ANTÔNIO JOSÉ BERUTTI VIEIRA

FENÔMENOS TERRESTRES COMO CLASSES DE OBJETOS: UM MODELO ESPAÇO-TEMPORAL

Tese apresentada ao Curso de Pós-Graduação em Ciências Geodésicas, Setor de Ciências da Terra, Universidade Federal do Paraná, como requisito parcial à obtenção do título de Doutor em Ciências Geodésicas.

Orientador:
Prof. Carlos Alberto Picanço de Carvalho, Ph.D.
Co-orientadora:
Prof.ª Dr.ª Cláudia Robbi Sluter

CURITIBA

2004

TERMO DE APROVAÇÃO

ANTÔNIO JOSÉ BERUTTI VIEIRA

FENÔMENOS TERRESTRES COMO CLASSES DE OBJETOS: UM MODELO ESPAÇO-TEMPORAL

Tese nº 035 aprovada como requisito parcial do grau de Doutor no Curso de Pós-Graduação em Ciências Geodésicas, Setor de Ciências da Terra, da Universidade Federal do Paraná, pela Comissão formada pelos professores:

Orientador: Prof. Dr. Carlos Alberto Picanço de Carvalho
Presidente (UFPR)

Prof. Dr. Luiz Alberto Vieira Dias
Membro (UNIVAP)

Prof. Dr. Leonardo Castro de Oliveira
Membro (IME)

Prof. Dr. Hélio Pedrini
Membro (UFPR)

Profa. Dra. Laura Sánchez Garcia
Membro (UFPR)

Prof. Dr. Henrique Firkowski
Membro (UFPR)

Curitiba, 28 de agosto de 2004.

Este trabalho é dedicado a minha esposa Marli Custódio Berutti Vieira e as minhas filhas Maíra e Rana.

AGRADECIMENTOS

Desejo externar meu profundo agradecimento ao professor Carlos Alberto Picanço de Carvalho pela orientação desta tese. Agradeço pelas intermináveis discussões e incentivos durante toda a nossa trajetória.

Desejo agradecer à professora Cláudia Robbi Sluter pela incansável disposição para me ouvir e pelas sugestões enriquecedoras apresentadas.

Meus agradecimentos também:

A minha mulher e as minhas filhas, pela compreensão pelas horas em que estive ausente e pelo amor que me dedicam;

Ao professor Leonardo Castro de Oliveira do Departamento de Cartografia do Instituto Militar de Engenharia pelas considerações e sugestões apresentadas durante o desenvolvimento desta tese.

Aos colegas Henrique Firkowski, Hideo Araki e Luciene Delazari, que sempre tiveram próximos compartilhando e apresentando sugestões.

Ao professor Neilor Camargo pela sua cooperação que tanto ajudou a superar obstáculos em programação.

Ao professor Hélio Pedrini, do Departamento de Informática da UFPR, pelas grandes sugestões apresentadas.

Ao professor Pedro Faggion pelas palavras de incentivo e compreensão.

A secretária do Curso de Pós-Graduação em Ciências Geodésicas Verali Mônica Kleuser.

Ao Curso de Pós-Graduação, ao Departamento de Geomática e a todos os meus colegas de trabalho que direta ou indiretamente colaboraram para que realizasse esta tese.

“Eu só peço a Deus
Um pouco de malandragem
Pois sou criança
E não conheço a verdade
Eu sou poeta e não aprendi a amar
Eu sou poeta e não aprendi a amar”

Cazuza e Frejat

SUMÁRIO

LISTAS DE TABELAS

LISTAS DE FIGURAS

RESUMO

Dados espaciais são fundamentais na produção de cartas topográficas e para a construção de sistemas de informação geográfica. Dados espaciais apresentam duas componentes. Uma de caráter semântico e outra espacial. Normalmente, estes dados são tratados como invariantes no tempo e o elemento temporal, quando considerado, é encontrado somente no contexto semântico ou temático. Nesta Tese é proposto um modelo espaço-temporal para representar fenômenos terrestres que são caracterizados por dados espaciais. Duas questões conceituais estão sendo consideradas. A primeira diz respeito à parte espacial. A formalização do que é um fenômeno terrestre, a sua caracterização com base em dados espaciais, e quais são os fenômenos terrestres a serem modelados. A segunda questão diz respeito à parte temporal e está relacionada com a maneira de tratar a variação que tais fenômenos podem apresentar com o tempo. O modelo proposto tem como premissa que qualquer fenômeno terrestre tem uma semântica definida e é caracterizado espacialmente por sua superfície externa. O modelo é baseado em classes e objetos. O núcleo do modelo é constituído por três classes: "FenomenoTerrestre"; "AreaLevantamento"; e "Recobrimento". Todo e qualquer fenômeno é representado como um objeto da classe "FenomenoTerrestre". A classe "AreaLevantamento" é usada para acondicionar todos os fenômenos levantados numa certa época e a classe "Recobrimento" é voltada para representar a variação temporal, que é determinada comparando-se os dados atribuídos aos correspondentes atributos de cada objeto. Para expressar o modelo é utilizada a UML (*Unified Modeling Language*). Para validar o modelo foram realizados testes em que a superfície topográfica foi submetida a uma variação temporal para quatro épocas diferentes. Com os testes realizados ficou demonstrado que o modelo é adequado segundo os objetivos considerados. Os dados para os testes foram simulados a partir de uma carta topográfica digital do Centro Politécnico da Universidade Federal do Paraná. Para a visualização dos resultados foi utilizado o padrão VRML (*Virtual Reality Modeling Language*) e as classes GeoVRML.

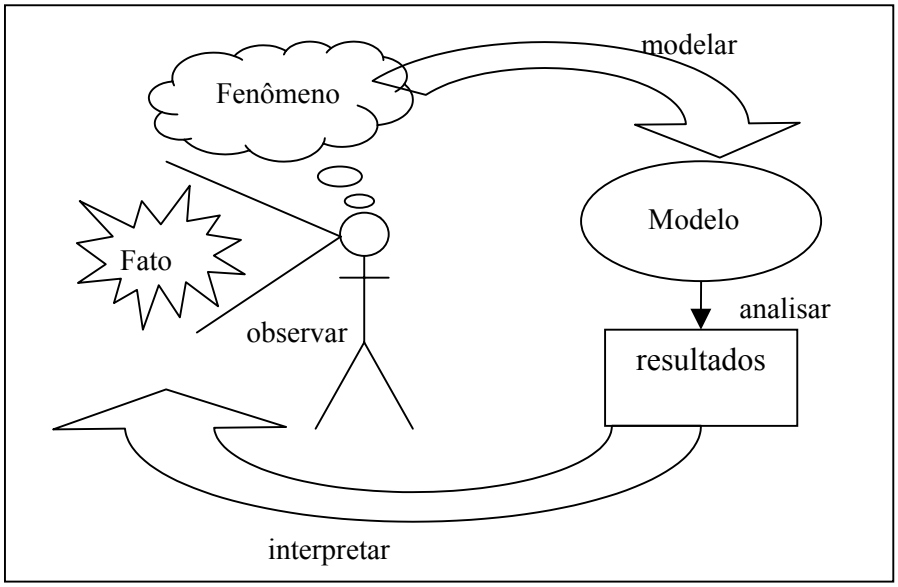
ABSTRACT

Spatial data are fundamentals to make topographic maps and to construct geographic information systems. These data have two components. One is semantic character and the other is spatial. Normally, spatial data are considered as invariants in time and the temporal element is found on semantic context. This Thesis presents a proposal for a spatio-temporal model which represents Earth´s phenomena characterized by spatial data. Two conceptual questions are addressed. One is concerning with spatial component. How to formalize Earth´s phenomena, how to characterize phenomena with spatial data, and what phenomena to be modeled. The second question is related with temporal component. How to represent temporal variation which phenomena should exhibit with the time. The proposed model has a premise that every Earth´s phenomena have a defined semantic and are characterized spatially by their external surfaces. The model is based on classes and objects. The core of the model is formed by three classes: "FenomenoTerrestre", "AreaLevantamento" and "Recobrimento". Every phenomenon is represented as an object of the class "FenomenoTerrestre". The "AreaLevantamento" class is used to pack all phenomena surveyed in a certain epoch. "Recobrimento" class represents the time variation, which is determined by matching the object´s attribute data. The proposed model is expressed by the Unified Modeling Language (UML). In order to validate the model some tests have been run. The topographic surface have been submitted a temporal variation to four different epochs. With the tests have been demonstrated that the proposed model is proper and in accordance with objectives. Data for these tests were simulated from a digital topographic map of the "Centro Politécnico da Universidade Federal do Paraná". Virtual Reality Modeling Language (VRML) combined with GeoVRML classes was used to visualize the temporal variation of the topographic surface and some phenomena.

1 INTRODUÇÃO

Nem sempre existe concordância entre os autores sobre o significado de certos termos que são envolvidos no processo de estudo de um problema. Por esta razão é apresentada na fig. 1 uma esquematização de alguns desses termos e suas relações, como estão sendo considerados neste trabalho. O termo *fato* está sendo definido como algo que existe no mundo real independentemente de alguém observá-lo ou não. Por sua vez, *fenômeno* é o entendimento que um observador tem de um fato e para isto existe um processo intelectual que depende do ponto de vista do observador. Visando compreender um fato, ou resolver um certo problema associado com este fato, adota-se um *modelo* para representar o fenômeno. A partir da análise realizada sobre o modelo são gerados *resultados* que após serem interpretados permitem inferir sobre o fato, desde que o modelo e o fenômeno sejam apropriados em relação ao fato real (VIEIRA; CARVALHO; SLUTER; 2002).

FIGURA 1.1 SEQÜÊNCIA DE ETAPAS NO PROCESSO DE ESTUDO DE UM FATO



A maneira como um modelo descreve um certo fenômeno está relacionada com os tipos de características do fenômeno representadas no modelo. Por exemplo, independentemente do tipo de modelo geométrico adotado para representar a forma da Terra, este modelo pode responder somente aquelas consultas de caráter espacial, tais

como: a posição, a forma ou a orientação. Quando se quer fazer referência a um modelo que permita consultas não somente sobre questões espaciais (“onde”), mas também sobre questões relacionadas com a semântica (“o que é”) e com a temporalidade (“quando”)**,** tem-se que usar um modelo que incorpore mais características (PEUQUET, 1999, p. 93). A denominação *modelo espaço-temporal* é, atualmente, utilizada para identificar um modelo em que estão representadas características**:** semânticas, espaciais e temporais (PEUQUET, 1999, p. 96).

A conseqüência natural da busca por soluções de problemas cada vez mais complexos faz com que os modelos correspondentes evoluam, no mínimo, na mesma proporção que os problemas. A situação comum neste processo de evolução dos modelos e dos sistemas computacionais, que são utilizados para a sua manipulação, é a criação de um novo modelo e o desenvolvimento de um novo sistema computacional que implemente o modelo. Isto, entretanto, pode implicar na perda total ou parcial do sistema existente. O desejável então é partir de um modelo**,** que represente a essência do fenômeno**,** e de um sistema que possam ser estendidos com o tempo para incorporar outras especializações. Tais sistemas são tratados em programação orientada a objetos por meio de conceitos tais como**:** classes, objetos, especialização, herança, polimorfismo e outros (BOOCH , 1994, p. 146). Alguns ensaios teóricos sobre a representação de dados espaciais foram feitos por VIEIRA e CARVALHO (2001b).

Em resposta a evolução dos modelos usados para dados espaciais, cada vez mais se busca formalismo para descrevê-los. Neste sentido, MOLENAAR (1998, p. 2) usa o termo *Teoria de Geo-Informação* para destacar o rigor matemático que vem-se buscando para formular os conceitos relacionados com dados espaciais. Do ponto de vista da ciência da computação, uma abordagem corrente é se adotar uma linguagem de modelagem. Com esta o modelo e o sistema podem ser expressos por um conjunto de regras com sintaxe, semântica e uma notação simbólica própria (ERIKSSON; PENKER, 1998, p. 28). Como conseqüência, há mais flexibilidade para formulações, análises e reformulações tanto do modelo, como do sistema.

A superfície terrestre tem um papel fundamental para todos os seres vivos e, em particular, para a espécie humana, uma vez que a maioria das atividades realizadas

pelo homem desenvolve-se sobre esta superfície. Então, a sua compreensão é de fundamental importância tanto para fins práticos, como para estudos científicos. Além disto, o entendimento da sua natureza pode diretamente contribuir para a compreensão de uma série de fenômenos correlacionados (HUTCHINSON, 1999, p. 105), e que são estudados em outras áreas de conhecimento como, por exemplo, hidrologia, morfologia, climatologia para citar somente algumas.

Do ponto de vista cartográfico, existem dois produtos que representam a superfície terrestre: a carta topográfica; e a carta náutica. Enquanto a carta náutica tem a sua ênfase na representação da superfície que está submersa pelos corpos hídricos, a carta topográfica tem a sua ênfase na representação da superfície que não está submersa. No contexto desta tese, adotou-se o termo superfície terrestre para considerar a parte externa da Terra**,** sobre a qual podem ocorrer e serem observados distintos fatos. Dentre os fatos que podem ser observados, os corpos hídricos são fenômenos importantes porque dividem esta superfície em duas regiões, ou seja, uma que está submersa e uma que não está submersa (VIEIRA et al., 2003). Para esta última se está denominando de superfície topográfica e sobre esta superfície está a maior parte dos fenômenos terrestres que são apresentados nas cartas topográficas.

Em geral, o conteúdo das cartas topográficas diz respeito com todos os fenômenos identificáveis sobre a superfície topográfica, se natural ou artificial, que podem assumir uma posição específica em relação a um sistema de coordenadas geodésicas. De acordo com KEATES (1980, p. 19) as informações representadas em cartas topográficas são relativas ao relevo, à posição planimétrica, ao meio físico, ao meio humano e aos nomes dos lugares. A importância das cartas topográficas reside no fato de que estas são produzidas para atender as necessidades governamentais em seus diferentes níveis. Além disto, muitas das informações representadas nestas cartas são utilizadas como um elemento de referência (ou elemento de fundo) para a produção dos outros tipos de cartas, também chamadas temáticas ou não topográficas. Assim, a ênfase que se está colocando no modelo proposto é com relação à representação de fenômenos terrestres apresentados nas cartas topográficas.

A mais de uma década que se vem tentando consolidar a pesquisa em

cartografia, dentro do Departamento de Geomática da Universidade Federal do Paraná. Dois são os principais problemas identificados para se atingir esta finalidade. Em primeiro lugar, percebeu-se que não existia uma base conceitual expressa de maneira formal e consistente sobre cartografia topográfica. Além disto, ficou evidente que grande parte dos esforços que vinham sendo realizados com o desenvolvimento de projetos de pesquisa e a elaboração de dissertações não estavam sendo incorporados de maneira a formar uma base comum de conhecimento que pudesse ser compartilhada e, gradativamente, fosse crescendo.

Quase no final da década de 90, o Curso de Pós-Graduação em Ciências Geodésicas começa a enfrentar o desafio de consolidar a pesquisa na área de cartografia. Para isto, são ministradas novas disciplinas, iniciam-se programas de tese e o desenvolvimento de novos projetos de pesquisa. Com o propósito de se estabelecer uma finalidade comum ao grupo de cartografia, ficou decidido que se deveria buscar o desenvolvimento de um sistema próprio, porque, com isto, se teria que dominar e expressar formalmente os elementos conceituais da área de cartografia, assim como, gerar um produto capaz de canalizar grande parte dos esforços realizados pelo grupo de pesquisadores.

1.1 DEFINIÇÃO DO PROBLEMA

A principal questão a ser considerada nesta tese está relacionada com a concepção de um modelo formal capaz de representar fenômenos terrestres quanto à semântica, espacialidade tridimensional e a evolução temporal. A principal dificuldade para chegar-se a isto é que tais fenômenos envolvem conceitos e idéias que muitas vezes são ambíguas quando expressas textualmente e, portanto, são difíceis para se estabelecer uma representação apropriada dentro de uma perspectiva formal. A hipótese assumida nesta tese é que é possível se conceber um modelo formal para representar fenômenos terrestres sob o ponto de vista da cartografia topográfica, tal que este modelo seja geral e consistente em relação ao mundo real, mas que ao mesmo tempo permita a sua extensão para representar novos fenômenos. Para isto, tem-se que definir conceitualmente o que é um fenômeno terrestre sob o ponto de vista da cartografia topográfica e estabelecer o conjunto de premissas sobre as quais o modelo

será concebido e expresso por meio de um conjunto de classes, que possam ser especializadas e combinadas para formar novas classes e expressar novos relacionamentos.

Até o momento, o aspecto temporal para fenômenos terrestres não é considerado na maioria dos sistemas computacionais. O mais comum é encontrar-se em alguns sistemas recursos de animação para visualização e neste caso, tem-se que criar uma cena, manualmente, compondo-a quadro a quadro. Uma comprovação de que o aspecto temporal ainda é considerado como uma questão aberta para pesquisa é a inexistência de recomendações por parte do *OpenGIS Consortium*, que é a referência internacional no que diz respeito ao estabelecimento de especificações e documentos que permitam interoperabilidade de tecnologias de geoprocessamento. Na documentação divulgada por esta entidade os fenômenos são assumidos como não tendo modificação com o tempo (OpenGIS Consortium, 2004).

1.2 OBJETIVO GERAL

Propor um modelo formal para representar fenômenos terrestres sob o ponto de vista da cartografia topográfica. O modelo deve expressar as características semânticas e espaciais tridimensionais dos fenômenos. Além disto, deve considerar que tais características podem variar com o tempo. Tal modelo deve ser geral e representar a essência dos fenômenos terrestres, mas ao mesmo tempo permitir sua extensão para expressar outros fenômenos que são especializados ou resultado de uma combinação de fenômenos existentes.

1.3 OBJETIVOS ESPECÍFICOS

A realização do objetivo geral pressupõe um conjunto de necessidades que têm de ser satisfeitas, a saber:

a) desenvolver a base conceitual capaz de fundamentar o modelo proposto;
b) identificar os tipos de fenômenos terrestres que serão modelados para compor um conjunto essencial de classes, a partir das quais outras classes possam ser derivadas;
b) identificar e propor termos para descrever os fenômenos terrestres como

classes de objetos;

c) definir, especificar e implementar um conjunto de classes, operações e relacionamentos por meio de uma linguagem de modelagem;

d) avaliar experimentalmente o modelo proposto a partir de dados simulados, tendo por base uma restituição digital tridimensional do Centro Politécnico da Universidade Federal do Paraná, na escala de 1:2000.

1.4 RELEVÂNCIA DO ASSUNTO

Do ponto de vista conceitual, a modelagem de fenômenos terrestres e a representação da variação temporal são tópicos relevantes para pesquisa científica. Ainda hoje, são poucos os autores que apresentam formalizações e resultados concretos para expressar tais fenômenos terrestres sob o ponto de vista da cartografia topográfica, que é a base para representar as outras formas de mapeamento. Acredita-se assim, que o maior benefício que se pode obter com a pesquisa aqui proposta é se chegar a essa fundamentação conceitual. A importância disto está em se eliminar qualquer grau de condicionamento do processo de pesquisa e de expressão do conhecimento na área de cartografia topográfica, além de possibilitar o desenvolvimento de um sistema próprio.

1.5 DESCRIÇÃO DOS CAPÍTULOS

A organização deste trabalho se dá segundo seis capítulos. O primeiro capítulo é uma breve introdução ao assunto e uma definição do problema que envolve a proposição de um modelo para representar fenômenos terrestres como classes de objetos. Está sendo considerando que o modelo deve contemplar características semânticas e espaciais tridimensionais e que os objetos representados podem sofrer variações com o tempo. No segundo capítulo é apresentada a revisão da literatura destacando a evolução dos modelos usados para representar dados espaciais. Em seguida são apresentados diferentes conceitos relacionados com a característica temporal e a busca por rigor matemático para expressar os modelos. Por último são descritos de forma resumida: a linguagem de modelagem UML, os principais conceitos relacionados com programação orientada a objetos, a linguagem de

programação *Java*, o padrão VRML, o pacote de classes GeoVRML e a linguagem HTML. No terceiro capítulo é apresentado o modelo proposto e são descritos as classes e os seus membros. É apresentado também como se determina a variação temporal para o caso da superfície topográfica. No quarto capítulo são apresentados os materiais utilizados no desenvolvimento da tese. No quinto capítulo são descritos os experimentos e as análises realizadas. Finalmente, no sexto capítulo são apresentadas as conclusões obtidas e as recomendações visando futuros trabalhos.

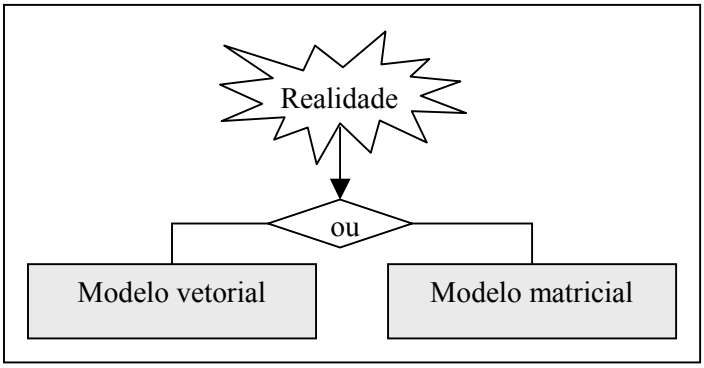
2 REVISÃO DE LITERATURA

2.1 EVOLUÇÃO DOS MODELOS DE DADOS ESPACIAIS

Os modelos usados para representar dados espaciais, com base em sistemas computacionais, surgiram antes da década de 80. MAGUIRE e DANGERMOND (1991, p. 320), afirmam que "... criar um modelo de dados envolve amostrar o espaço contínuo da realidade e representá-lo numa forma discreta". Os modelos mais comuns eram chamados: modelo vetorial e modelo matricial (BURROUGH, 1986, p.19; PEUQUET, 1984, p. 75). De acordo com BURROUGH (1986, p. 14) tais modelos representam a estrutura espacial do fenômeno. Daí serem, normalmente, referenciados como modelos de dados espaciais, em contraposição aos modelos de dados que detêm apenas informação semântica.

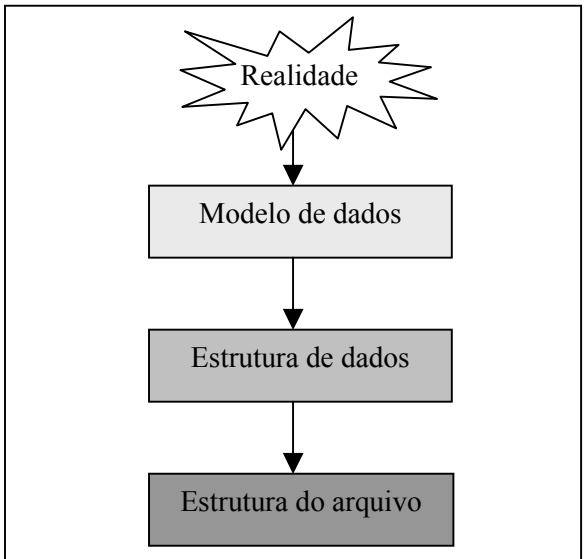
Por algum tempo os modelos vetorial e matricial condicionaram os tipos de sistemas de informação geográfica, ou seja, existiam sistemas que eram voltados para lidar somente com estruturas matriciais enquanto que existiam outros que eram voltados para estruturas vetoriais. Quando os dados espaciais eram coletados com equipamentos topográficos e geodésicos, com restituidores fotogramétricos ou com mesas de digitalização cartográfica, se usava o modelo vetorial para representá-los. Se os dados espaciais fossem originados de imagens coletadas por satélites artificiais ou escanerização de mapas, se usava o modelo matricial (fig. 2).

FIGURA 2 - PRIMEIRO MODELO PARA REPRESENTAR DADOS ESPACIAIS



Posteriormente, PEUQUET (1984, p. 69) apresenta uma hierarquia de abstração para classificar os modelos. Quanto mais próximo o modelo está da realidade, maior é o grau de abstração. Quanto mais próximo o modelo está dos dispositivos físicos, menor é o grau de abstração. Do ponto de vista da aplicação se procura ficar mais próximo da abstração, enquanto que do ponto de vista da implementação se busca ficar mais próximo dos dispositivos físicos. De acordo com PEUQUET (1984, p. 69), a partir da *realidade* os modelos de dados espaciais são distribuídos em três níveis de abstração. O primeiro nível é chamado de *modelo de dados* - que é o nível mais alto de abstração. Neste modelo são incorporadas aquelas propriedades que se imagina que sejam relevantes para uma aplicação em mente. O segundo modelo é chamado de *estrutura de dados* - que é uma representação do modelo de dados na forma de diagramas, listas e matrizes. O terceiro modelo é chamado de *estrutura do arquivo* - e representa a estrutura de dados nos dispositivos de armazenamento (fig. 3).

FIGURA 3 - NÍVEIS DE ABSTRAÇÃO PARA OS MODELOS DE DADOS

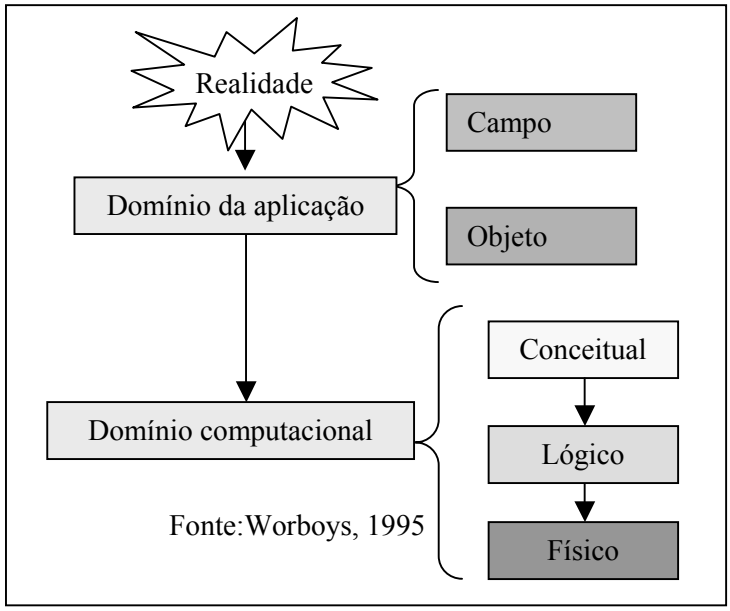


Fonte: Maguire e Dangermond, 1991

Uma abordagem alternativa para classificar os modelos é apresentada por WORBOYS (1995, p.147), que subdivide os modelos em relação ao domínio da aplicação e ao domínio computacional. Para o domínio da aplicação tem-se o *modelo campo* e o *modelo objeto*, que é como atualmente a maioria dos autores também

relatam (MOLENAAR, 1998, p. 1; LAURINI; THOMPSON, 1998, p. 87). Para o domínio computacional, têm-se os níveis: conceitual, lógico e físico. O destaque a ser feito sobre o modelo de WORBOYS é com relação à separação entre os domínios e a introdução dos modelos campo e objeto (fig. 4).

FIGURA 4 - NÍVEIS DE ABSTRAÇÃO PROPOSTO WORBOYS



A escolha entre o modelo campo e modelo objeto, no domínio da aplicação, depende de como o usuário entende o fenômeno que está para ser representado. Por exemplo, se o usuário entende o fenômeno espacial como sendo um padrão cuja distribuição espacial é contínua, ou seja, com características de um campo sobre uma região da superfície terrestre (a idéia de campo físico), então o modelo campo é o mais indicado (BONHAM-CARTER, 1994, p. 27). Exemplos para isto podem ser, os casos de variações de temperatura, de pressão ou de gravidade. Por outro lado, se o usuário entende o fenômeno espacial como algo distinto, ou seja, uma entidade isolada, então o modelo objeto é o mais adequado, como por exemplo, no caso de uma casa, um rio ou uma rodovia. Segundo COUCLELIS[1], apud WORBOYS (1995, p. 149), é possível fazer um paralelo destes tipos de modelos e o dilema filosófico, do início do século 20, entre o *pleno* e o *atômico*, ou então a *partícula* (modelo objeto) versos a *onda* (modelo

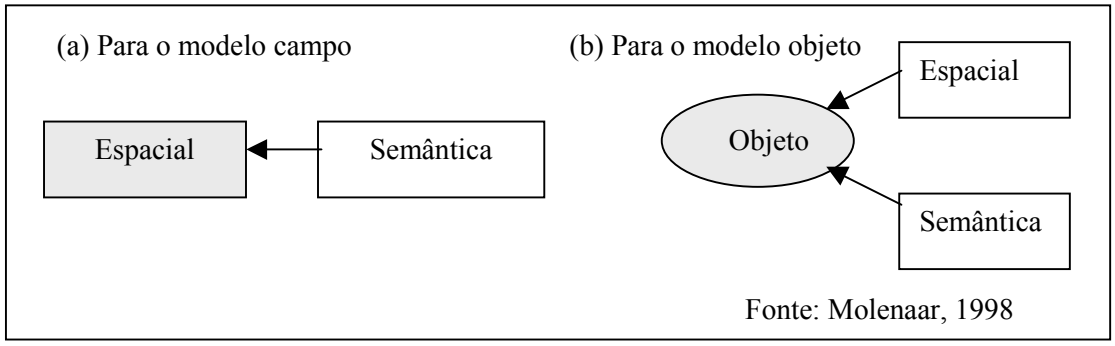
[1] COUCLELIS, H. Beyond the raster-vector debate in GIS. In.: FRANK, A. U.; CAMPARI, I. FORMENTINI, U. **Theories of Spatio-Temporal Reasoning in Geographical Space, Lecture Notes in Computer Science** 639, Berlin: Spring-Verlag, p.65-77. 1992.

campo).

Existem situações em que não se pode aceitar a hipótese de que os fenômenos sejam modelados como campo ou como objeto, ou seja, embora os fenômenos tenham limites, estes são nebulosos. Esta situação é comum quando os limites do fenômeno se distribuem sobre uma faixa ou região. Um exemplo típico diz respeito ao mapeamento e a classificação de tipos de solo ou de vegetação. De acordo com FISHER (1999, p. 193) estes fenômenos são caracterizados por limites pobremente definidos. Para os estudos que se seguem, "fenômenos nebulosos" não estão sendo considerados porque estes não fazem parte da cartografia topográfica, que é o principal escopo deste trabalho.

Apesar dos dados espaciais conterem características semânticas e espaciais, normalmente são referenciados apenas como dados espaciais. De acordo com MOLENAAR (1998, p. 4), os dados espaciais são chamados também de informação geométrica para diferenciar da informação temática (a denominação temática aqui é usada por MOLENAAR como um sinônimo de semântica). É importante destacar que dependendo do modelo que se adote, a forma com que as informações se vinculam é diferente (MOLENAAR, 1998, p. 5), ou seja, para o modelo campo a informação semântica está diretamente vinculada com a informação espacial (fig. 5a). Entretanto, para o modelo objeto as informações semânticas e espaciais se vinculam por meio do objeto, ou do identificador do objeto (fig. 5b). Este tipo de vinculação pode ser necessário quando se deseja fazer algum tipo de busca ou consulta aos dados.

FIGURA 5 - VINCULAÇÃO DAS INFORMAÇÕES PARA MODELO: (a) CAMPO, (b) OBJETO



2.2 CARACTERÍSTICA TEMPORAL

Um tema atual com relação à evolução dos modelos de dados espaciais é a incorporação da característica temporal para os fenômenos terrestres. Após quase três décadas de existência de bases de dados espaciais e de sistemas de informação geográfica, PEUQUET (1999, p. 91) afirma que as representações usadas nessas bases de dados ainda assumem que o mundo real existe somente para um único instante. Com isto, tem-se um mundo real estático, o que não é verdade. Hoje os usuários vêm sendo desencorajados a descartar os dados ditos desatualizados quando novos dados são coletados, procedimento que é conhecido como *agony of delete* (COPELAND[2] e LANGRAN[3], apud PEUQUET, 1999, p. 91). Ao concluir seu artigo, PEUQUET (1999, p. 101) enfatiza que "... ainda resta muito a ser feito antes que um Sistema de Informações Geográficas verdadeiramente temporal possa ser realizado".

Quando se considera o ciclo de vida da informação associada a um fenômeno terrestre, existem quatro fases que são: a coleta, o processamento, o uso e a atualização. A fase de coleta do dado envolve duas etapas: medir e armazenar. O custo e o tempo relativamente altos para medir os dados e a baixa capacidade dos meios de armazenamento podem ser apontados como os fatores mais restritivos para a incorporação da característica temporal aos modelos usados para representar os fenômenos terrestres.

Atualmente esse quadro restritivo se modificou, principalmente quanto à obtenção de posições sobre a superfície terrestre e de coberturas de imagens obtidas com sensores multiespectrais. Com o uso de receptores GPS (*Global Positioning System*) junto com rádios transmissores é possível coletar dados posicionais sobre a superfície terrestre em tempo real. Além disto, com o uso de sensores em satélites, particularmente do sensor TM (*Thematic Mapper*) transportado no satélite LANDSAT 5 é possível obter imagens de grandes áreas da superfície terrestre (170 x 180 km) com uma freqüência de 16 dias e uma resolução espacial de 30m (USGS, 2004b). Para que essas tecnologias atingissem o estágio atual foi necessário que a capacidade dos

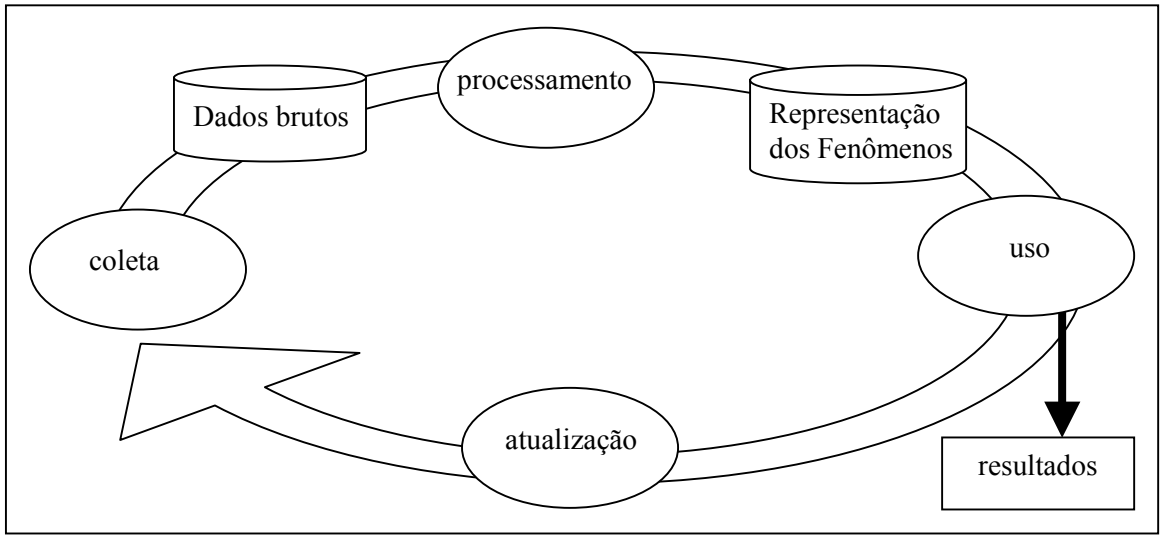
---

[2] COPELAND, G. What if mass storage were free ? **Computer** n. 15. P. 27-35.1982.
[3] LANGRAN, G. **Time in geographical systems.** London. Taylor & Fracis. 1992.

dispositivos de armazenamento de dados também evoluíssem. Hoje as unidades de disco e mídias digitais, como DVD (*Digital Versatile Disc*) e CD (*Compact Disc*), armazenam volume de dados respectivamente da ordem de 80GB, 4,7GB e 700MB e continuam evoluindo. Com isto, é possível se dizer que a fase de coleta de dados pode não ser o fator condicionante.

A fase de processamento pode ser decomposta nas etapas: modelar e persistir, que consiste em deixar disponível para uso. Pode-se dizer que esta fase apresenta as maiores limitações, porque depende do modelo usado para representar os fenômenos e isto tem influencia sobre à persistência, bem como as fases de uso e atualização. A fase de uso está ligada ao usuário final, que consulta as informações disponíveis no sistema. Além das restrições que podem ser impostas pelo modelo de representação, pode-se destacar que a disponibilidade de interfaces gráficas pode tornar esta interação mais fácil. Em resposta às necessidades da fase de uso, inicia-se a fase de atualização, que aciona as etapas relacionadas com a fase de obtenção de dados reiniciando, portanto, o ciclo (fig. 6).

FIGURA 6 - CICLO DE VIDA DA INFORMAÇÃO ASSOCIADA AOS FENÔMENOS TERRESTRES

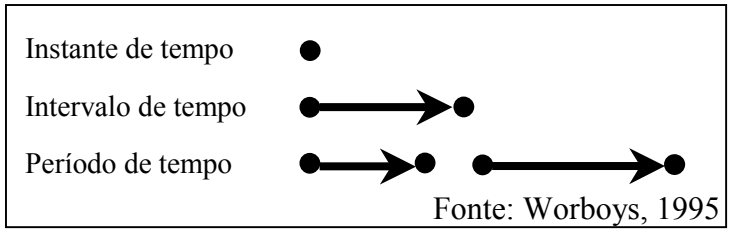


2.2.1 Medição e Associação da Característica Temporal

A forma usada para medir a característica temporal é instantânea. De acordo com WORBOYS (1995, p. 310), existem três formas de associação da característica

temporal ao fenômeno. A primeira é dita instantânea e o fenômeno está diretamente associado com o instante de tempo em que foi observado. A segunda forma de associação é aquela em que o fenômeno está associado a um intervalo de tempo. Neste caso, tem-se de medir o instante inicial e o instante final durante o qual ocorreu o evento (fig. 7). A terceira forma de associação é aquela em que o fenômeno está associado a um período de tempo, ou seja, um conjunto formado por um certo número de intervalos de tempo.

FIGURA 7 - ESTRUTURA TEMPORAL



Fonte: Worboys, 1995

2.2.2 Tipos de Tempo

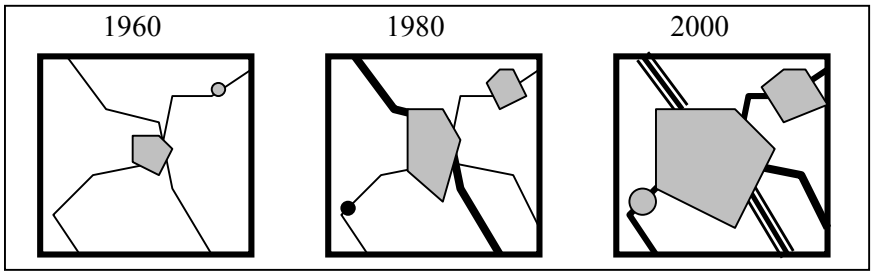
De modo geral, *tempo válido* é definido como sendo o instante em que um evento é observado no domínio da aplicação e *tempo de transação* é o instante em que uma transação se realiza no sistema, também denominado de *tempo de base de dados* ou *tempo de sistema*. (WORBOYS, 1995, p. 309; PRICE, SRINIVASAN, RAMAMOHANARAO, 1999, p. 171; JONES, 1997, p. 26). Neste ponto é possível constatar a influência da terminologia usada na época em informática para se referir aos dados comerciais que se alteravam com o passar do tempo.

Além do tempo válido e do tempo de base de dados, outros tipos são apresentados na literatura, como por exemplo, aquele em que o dado é recuperado da base de dados, que é denominado de *tempo de leitura* (PRICE, SRINIVASAN e RAMAMOHANARAO 1999, p. 171), ou de *tempo de referência* (WORBOYS, 1995, p. 311). PRICE, SRINIVASAN e RAMAMOHANARAO (1999, p. 171) destacam mais dois tipos: o *tempo de existência*, que tem sido usado para descrever o instante em que um objeto existe no mundo real, mas que inclui, necessariamente, o tempo válido de qualquer instância associada com ele. O outro é o tempo *definido pelo usuário*, cujo significado é estabelecido no contexto da aplicação.

Embora sejam encontradas na literatura muitas definições para os tipos de

tempo, não são apresentados os correspondentes exemplos demonstrando realmente os seus usos. Com relação a isto, supõe-se que, simplesmente, foi tentada uma adaptação da terminologia usada na época pela computação voltada para fins comerciais, mas que não são aplicáveis sob o ponto de vista da cartografia topográfica. O exemplo mais comum que é apresentado pelos autores é o uso da característica temporal na apresentação de uma animação cartográfica. Um tipo desta animação consiste na utilização de mapas que retratam diferentes épocas para as quais ocorreu uma variação temporal nos elementos espaciais. Para tanto, os mapas são gerados separadamente e organizados numa seqüência cronológica para formar uma animação. KRAAK e ORMELING (1996, p. 66) denominam este processo de quadro a quadro (fig. 8).

FIGURA 8 - IMAGENS REPRESENTANDO ÉPOCAS DISTINTAS



2.2.3 Tipos de Dados

Tendo por base as características espaciais, semânticas e temporais, PRICE, SRINIVASAN e RAMAMOHANARAO (1999, p. 164) sugerem a seguinte classificação: dados espaciais, dados temporais, dados espaço-temporais, dados semânticos e dados compostos. Segundo estes autores, os dados espaciais são aqueles que têm somente domínio espacial - por exemplo, a extensão da propriedade. Por sua vez, os dados temporais são aqueles que têm somente domínio temporal - tempo de vôo de uma aeronave. Os dados espaço-temporais são dados espaciais que se alteram com o tempo, como por exemplo, as mudanças com respeito à forma, ao tamanho, a posição ou a orientação. No que se refere aos dados semânticos, estes podem ter variações em conseqüência de mudanças puramente espaciais, ou temporais ou espaço-temporais e exemplificam citando o caso da acidez do solo, que pode variar de lugar para lugar e com o tempo. Dados compostos são aqueles constituídos por

associações de dados e cujos componentes podem mudar com o tempo ou espacialmente.

Pela classificação proposta por PRICE, SRINIVASAN e RAMAMOHANARAO (1999, p. 164), os dados espaciais e dados temporais não sofrem alteração, nem com o passar do tempo e nem espacialmente. Estes retratam um fenômeno de forma estática, tanto no espaço como no tempo. Daí esses autores afirmarem que somente para os três últimos tipos, a saber: dados semânticos, dados espaço-temporais e dados compostos, faz-se necessário armazenar tanto o valor inicial como as variações que ocorrem.

2.3 FORMALIZAÇÃO PARA EXPRESSAR OS MODELOS

Segundo PEUQUET (1984, p. 67), um modelo de dados "... é uma descrição geral de conjuntos específicos de entidades e das relações entre estes conjuntos de entidades". Em acréscimo, PEUQUET (1984, p. 69) relata que "... a característica mais básica de um modelo de dados é que ele é uma abstração da realidade". CODD[4] , apud PEUQUET (1984, p. 69) afirma que "... um modelo de dados consiste de três componentes: uma coleção de tipos de objetos, uma coleção de operadores e uma coleção geral de regras de integridade". WORBOYS (1995, p. 145) considera que um modelo "... é uma construção artificial na qual partes de um domínio (domínio fonte) são representadas num outro domínio (domínio alvo) (...) o propósito de um modelo é simplificar e abstrair o domínio fonte".

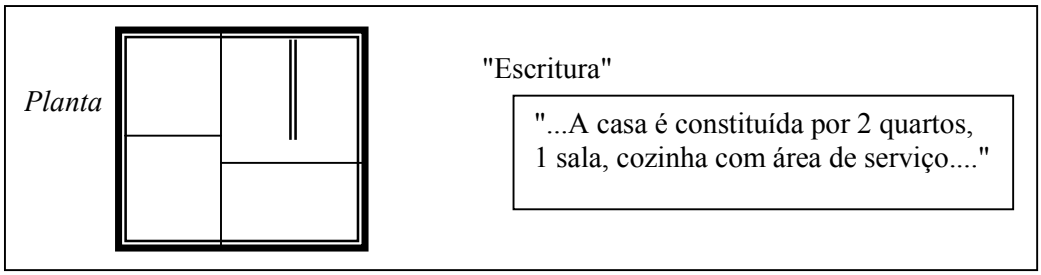
É interessante observar que, para as definições apresentadas no parágrafo anterior, existe uma intenção explícita dos autores na busca por formalismo para expressar os modelos. Isto se evidencia com o uso de termos e conceitos comuns da Álgebra, como por exemplo: conjuntos, entidades, relações e domínios. Pode-se dizer que isto é conseqüência de uma evolução natural na estruturação do conhecimento sobre os tipos de modelos de dados. Quanto ao estágio de desenvolvimento da modelagem de dados espaciais, tem-se inicialmente uma fase dita empírica, por estar apoiada num processo com bases intuitivas, mas que vem se transformando na busca

[4] CODD, E. F. A relational model of data for large shared data banks. Communications of the ACM, v. 13, p. 378-387. 1970.

por reformulação dos seus conceitos com rigor matemático. Para isto, MOLENAAR (1998, p. 2) usa a denominação de *Teoria de Geo-Informação*.

Mais recentemente, alguns autores estendem a definição de modelo fazendo referência à linguagem de modelagem. ERIKSSON e PENKER (1998, p. 7) afirmam que um modelo "... é uma descrição de alguma coisa e tem um propósito (...) um modelo é expresso por uma linguagem de modelagem (...) linguagem de modelagem consiste de uma notação - os símbolos usados nos modelos - e um conjunto de regras de como usá-la". Na fig. 9 são apresentados exemplos de dois modelos distintos que podem ser usados para descrever uma casa. A diferença está em que no modelo planta se usa uma linguagem gráfica e no modelo escritura se usa uma linguagem textual.

FIGURA 9 - EXEMPLO DE MODELOS DE UMA CASA



2.4 LINGUAGEM DE MODELAGEM UML

No que diz respeito aos desenvolvimentos correntes em computação pode-se destacar as linguagens de modelagem, em particular a UML (*Unified Modeling Language*). A UML é baseada no paradigma de orientação a objeto, portanto permite a conceituação do problema por meio de classes e objetos, e expressa os modelos por meio de grafos, chamados diagramas (ERIKSSON; PENKER, 1998, p. 5). Um outro aspecto importante é que o OMG[5] (*Object Management Group*) reconheceu a UML como um padrão formal (ALHIR, 1998, p. 13). Além disto, a UML contém conceitos novos que permitem expandi-la e adaptá-la para expressar padrões específicos definidos pelo usuário (ERIKSSON; PENKER, 1998, p. 28).

Como é relatado por ERIKSSON e PENKER (1998, p. 3), o que motivou a

[5] OMG é um consórcio aberto sem fins lucrativos que produz e mantém especificações para a indústria computacional que visa modelar o mundo real por meio de “objetos”.

concepção da UML foi propor algo que fosse capaz de se contrapor a "guerra de métodos" que se observava dentro da comunidade de programação orientada a objeto durante a década de 90. Numa primeira fase foi proposto um método que tinha por base os métodos de Booch e de Rumbaugh (*Object Modeling Technique - OMT*). Posteriormente, foi incorporado o método de Jacobson (*OOSE e o Objectory*). Outros métodos também foram incluídos, como o método *Fusion,* da Hewlett-Packard, e o método *Coad/Yourdon*.

Os desenvolvedores da UML entenderam que a proposta de um método único seria inviável e propuseram uma linguagem de modelagem, que foi denominada de "Linguagem de Modelagem Unificada" (ERIKSSON; PENKER, 1998, p. 5). A diferença básica entre uma linguagem e um método é que o método diz o que fazer, como fazer, porque fazer e quando fazer e, portanto, isto tem implicações diretas com uma organização em função de objetivos a serem atingidos (ERIKSSON; PENKER, 1998, p. 7). Ao contrário, uma linguagem consiste de notação, regras, semântica e sintaxe para expressar os modelos que compõem o método.

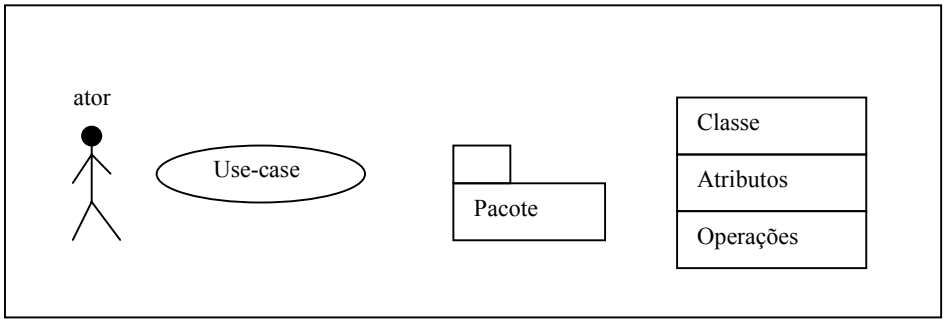
A versão atual da UML é a 1.3 (OMG, 2001). Quando se considera a sua evolução (ALHIR, 1998, p. 9), quatro épocas podem ser destacadas:

a) fragmentação - a "guerra dos métodos";
b) unificação - o esforço de unificação de métodos ao nível de linguagem;
c) padronização - a versão 1.1 é submetida a OMG em 09/1997;
d) industrialização - a OMG adota a UML como padrão formal em 11/1997.

Uma questão chave na UML é a comunicação. Isto está baseado na premissa de que se a comunicação é possível durante o esforço de trabalho para solucionar problemas, é provável que o trabalho e o conhecimento sejam compartilhados e o esforço reduzido (ALHIR, 1998, p. 16). Neste sentido, a UML é dita uma linguagem para visualização (PRICE, SRINIVASAN, RAMAMOHANARAO, 1999, p. 14). Uma outra propriedade importante na UML é criar modelos. De acordo com ERIKSSON e PENKER (1998, p. 2) para serem considerados úteis os modelos devem contemplar aspectos, tais como: ser simples; acurado; consistente; fácil de comunicar e fácil para mudar.

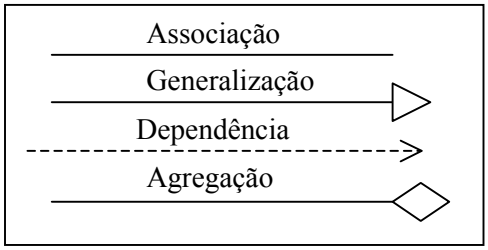
Para lidar com esses aspectos a UML dispõe de elementos de modelo, diagramas e vistas (ERIKSSON; PENKER, 1998, p. 13). Os elementos de modelos são os conceitos representados nos diagramas, sendo que os elementos de modelo têm uma semântica própria, uma definição formal e uma representação gráfica única. Na fig. 10 são apresentados alguns dos principais elementos de modelos da UML. Um ator é alguém ou alguma coisa que interage com o sistema . Um caso de uso representa uma funcionalidade completa como percebida pelo ator. Um pacote é uma forma de agrupar elementos que têm afinidade entre si. A classe é usada para modelar todas as coisas do sistema.

FIGURA 10 - ALGUNS ELEMENTOS DE MODELOS DA UML



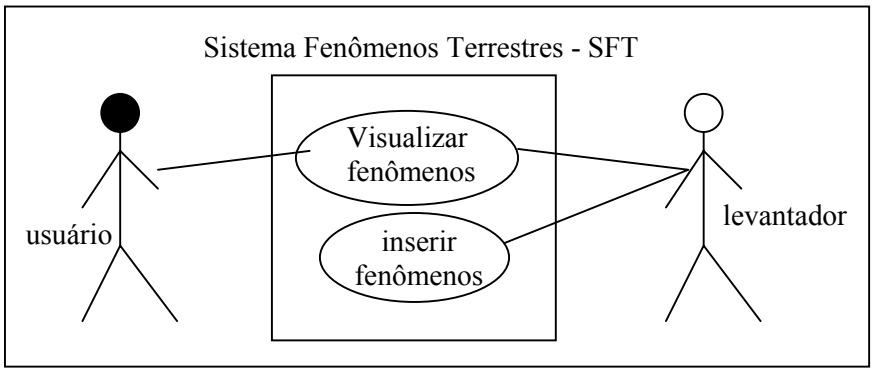
Existem outros elementos de modelo que são usados para representar os relacionamentos entre os elementos, como por exemplo: associação - que conecta elementos e vincula as instâncias; generalização - um elemento é uma especialização de outro elemento; dependência - um elemento depende de outro elemento; e agregação - um tipo associação em que um elemento contém outros elementos (fig. 11).

FIGURA 11 - EXEMPLOS DE ALGUNS TIPOS DE RELAÇÕES



Os diagramas são grafos que mostram os elementos de modelo arranjados apropriadamente para ilustrar um aspecto particular do sistema (ERIKSSON; PENKER, 1998, p. 17). Diagrama casos de uso (*use-cases diagram*) é um diagrama que descreve uma ou mais funcionalidades que o sistema proporciona, mas com um enfoque externo de como se imagina que o sistema proporcione essa funcionalidade, ou seja, não existe preocupação de como isto será realizado internamente no sistema. As vistas são abstrações, que apresentam aspectos particulares do sistema. Uma vista não é um grafo, mas é constituída por um certo número de diagramas (ERIKSSON; PENKER, 1998, p. 14). No diagrama casos de uso apresentado na fig. 12, tem-se um exemplo de elementos de modelo: ator (usuário e levantador); casos de uso (visualizar fenômenos e inserir fenômenos) e o Sistema para Fenômenos Terrestres.

FIGURA 12 - EXEMPLO DE UM DIAGRAMA CASOS DE USO



2.5 ALGUNS CONCEITOS EM PROGRAMAÇÃO ORIENTADA A OBJETOS

BOOCH (1994, p. 146) relata que uma das etapas mais difíceis no desenvolvimento de um sistema orientado a objetos é identificar classes e objetos. Isto depende tanto de um processo de descoberta como de invenção. Com o processo de descoberta é possível se reconhecer as abstrações que formam o vocabulário do domínio do problema. Com o processo de invenção se formulam abstrações generalizadas e mecanismos que especificam como os objetos colaboram entre si. Assim, cada fenômeno que tem de ser modelado é uma parte do vocabulário usado para caracterizar o problema ou a sua solução. De acordo com BOOCH, RUMBAUGH e JACOBSON (1999, p. 55), modelar o vocabulário de um sistema

consiste em:

a) identificar as coisas que são usadas para descrever o problema ou solução;
b) identificar o conjunto de responsabilidades para cada classe;
c) prover os atributos e operações que são necessários para realizar as responsabilidades de cada classe.

A elaboração de um sistema complexo envolve a construção de vários modelos e passa pela aplicação sistemática de três princípios, que são: a decomposição, a abstração e a hierarquização (BOOCH, 1994, pg. 23). Tais princípios são conduzidos de maneira incremental e iterativa. BOOCH (1994, p. 83) afirma: "um objeto tem estado, comportamento e identidade.". KHOSHFIAN[6] e COPELAND, apud BOOCH (1994, p. 91) definem identidade como: "... aquela propriedade de um objeto que o distingue de todos os outros objetos". De acordo com BOOCH (1994, p. 86), o comportamento de um objeto: "... é como ele age e reage, em termos de passagem de mensagens e mudança de estado". BOOCH (1994, p. 84) afirma: "... o estado de um objeto inclui todas as propriedades (usualmente estáticas) do objeto e os valores correntes (usualmente dinâmicos) de cada uma das propriedades". Enquanto um objeto representa algo de concreto, porque seus atributos assumem valores específicos e ocupam espaço na memória do computador, uma classe representa a essência do objeto (BOOCH, 1994, p. 103).

2.5.1 Tipos de Classes

De acordo com FLANAGAN (2000, p. 91), uma classe "... é uma coleção de dados, armazenados em campos nomeados, e de código, organizado em métodos nomeados, os quais operam naqueles dados". Os campos e métodos recebem uma denominação particular, são chamados de membros da classe. Quanto aos tipos de membros de uma classe, FLANAGAN (2000, p. 92) relata os seguintes tipos: campo de classe; método de classe; campo de instância; método de instância. Um campo de classe (também chamado de campo estático) é um valor constante e único

[6] KHOSHAFIAN, S. and COPELAND, G. Object Identity. SIGPLAN Notices. Vol. 21(11), p. 406. Nov. 1986.

(FLANAGAN, 2000, p. 93), mesmo que um novo objeto seja criado. Um campo de classe funciona como uma variável global. Na sua declaração são utilizados os modificadores "*static*" e "*final*" para caracterizar que são campos que, após serem inicializados, não se modificam.

Semelhantemente aos campos de classe, os métodos de classe são métodos globais e também são declarados com o modificador "*static*". Estes métodos são também chamados de métodos utilitários, porque são usados de maneira útil, mas não operam sobre os objetos (ou instâncias) da classe. Com isto, não é necessário se instanciar um objeto para usar um método de classe.

Ao contrário dos métodos e campos de classe, os campos e métodos de instância estão associados ao objeto. Toda vez que um objeto é criado existirá uma cópia dos campos de instância para este objeto e os métodos são operados sobre o objeto criado.

No contexto da programação orientada, os conceitos classe, subclasse e herança estão intimamente relacionados. Por exemplo, ao se criar uma nova classe é possível fazer com que esta classe herde tanto os campos como os métodos de uma classe existente. Para a classe existente se usa denominar de superclasse, enquanto que para a classe derivada se chama de subclasse (DEITEL e DEITEL, 2001, p. 386). O relacionamento entre a superclasse e a subclasse é chamada de herança. Uma subclasse é mais específica do que a sua superclasse, porque esta pode ter adicionalmente seus próprios campos e métodos.

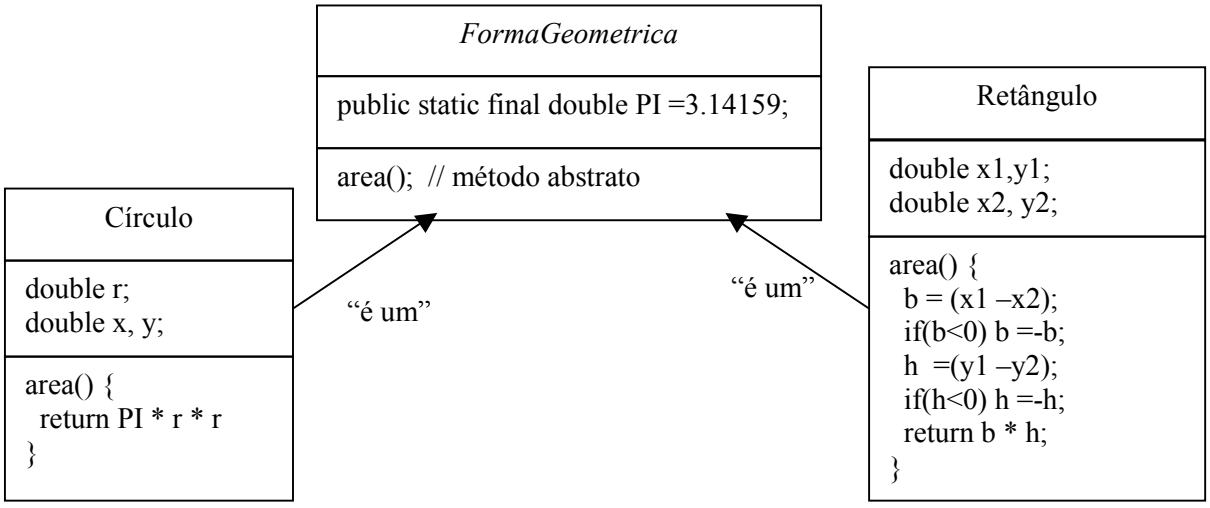
As classes podem ser classificadas em dois tipos: classe abstrata e classe concreta. De acordo com DEITEL e DEITEL (2001, p. 406), existem situações de programação em que não são realmente instanciados objetos de uma certa classe, mas esta classe é utilizada como uma superclasse dentro de uma hierarquia de classes. As classes deste tipo são chamadas de classes abstratas. Normalmente, essas classes são utilizadas como superclasses em situações de herança. Ao contrário, uma classe concreta é aquela para a qual os objetos são instanciados. As classes concretas fornecem os aspectos específicos que permitem a instanciação dos objetos.

Um outro conceito para lidar com complexidade de software é o mecanismo

de polimorfismo, que permite que se escrevam programas de uma forma geral para tratar uma ampla variedade de classes que têm afinidade. DEITEL e DEITEL (2001, p. 385) relatam que com o polimorfismo é fácil adicionar novos recursos a um sistema estendendo-se dessa forma o sistema. É importante destacar que uma das vantagens da programação orientada a objetos está na possibilidade de combinar vários desses conceitos. Por exemplo, na fig. 13 é apresentada uma classe abstrata chamada "*FormaGeometrica*" e neste caso esta é a superclasse, que possui um método abstrato "área()" e um membro de campo "PI". A partir desta classe são derivadas duas subclasses ("Circulo" e "Retângulo"), que são subclasses concretas de "*FormaGeometrica*".

Responsabilidade de uma classe diz respeito com as obrigações da classe. Na prática, uma classe bem estruturada deve ter no mínimo uma responsabilidade, mas não muitas (BOOCH; RUMBAUGH; JACOBSON, 1999, p. 53). Por exemplo, quando se imagina a classe forma geométrica se pode dizer que uma das suas responsabilidades é informar o valor da área, que é expresso pelo método área() (fig. 13).

FIGURA 13 - EXEMPLO DE POLIMORFISMO EXPRESSO COM UML



2.6 A LINGUAGEM DE PROGRAMAÇÃO *Java*

Como concebida pela *Sun Microsystems*, *Java* é considerada como sendo uma plataforma de linguagem de programação. De acordo com GOSLING e MCGILTON (2004), dentre as características importantes de *Java* destacam-se: é uma linguagem de programação orientada a objetos e tem um ciclo de desenvolvimento mais rápido, por ser uma tecnologia interpretada. Além disto, permite que os aplicativos *Java* sejam executados em diferentes sistemas operacionais e arquiteturas. Os aplicativos *Java* são considerados robustos, porque o ambiente *Java* gerencia a memória. Os aplicativos gráficos interativos têm alto desempenho, porque são suportadas atividades múltiplas e concorrentes. Os aplicativos são adaptáveis às mudanças de ambientes, porque é possível obter-se, dinamicamente, os módulos de código a partir da Web.

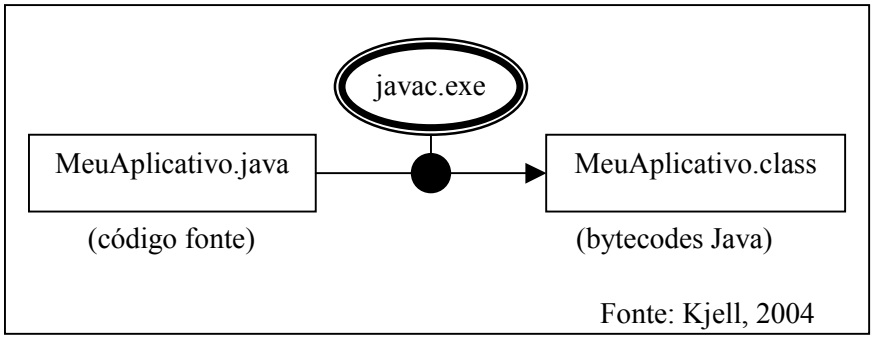
A principal motivação para a concepção de *Java* foi lidar com problemas de construção de software para dispositivos distribuídos. Com a popularização da Web as características de *Java* tornaram-se muito mais atraentes. Tendo *Java* como uma linguagem de extensão, é possível fazer com que os navegadores Web não fiquem restritos a um conjunto limitado de capacidades. Para isto, são embutidos em documentos HTML um tipo de programa específico chamado de *applet Java* (LINDHOLM; YELLIN, 2004).

Java é considerada como uma linguagem que pode ser mais facilmente aprendida, principalmente por quem tem experiência em linguagem de programação C ou C++ (GOSLING; MCGILTON, 2004). É possível se ter independência de plataforma quando se usa *Java* na sua especificação original. A distribuição de novas classes pode ser feita a partir de uma servidora central, em que essas classes podem ser carregadas sem a necessidade de recompilação de todo o programa (KJELL, 2004).

Existem duas maneiras de um sistema computacional rodar um programa fonte. A primeira é traduzir as linhas do arquivo com o código fonte em instruções de máquina (KJELL, 2004). A segunda alternativa é usar um programa interpretador. A linguagem *Java* combina estas duas idéias. Para rodar um programa em *Java*, o

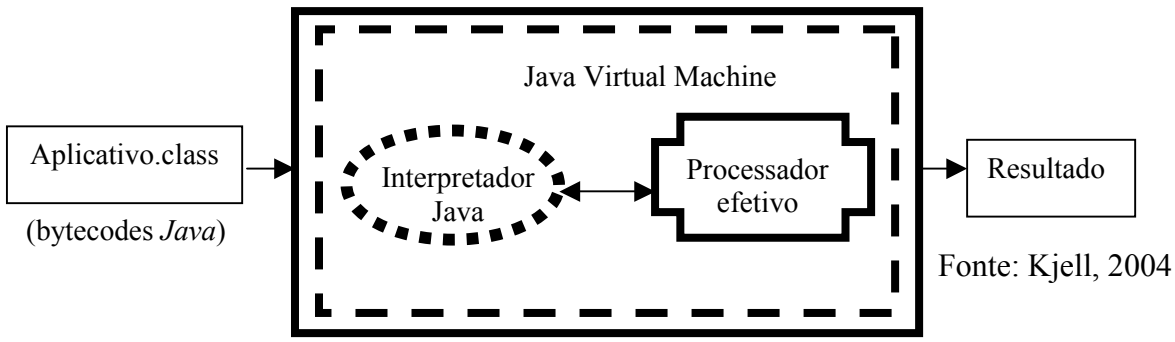
compilador *Java* ("javac.exe") traduz o código fonte para *bytecodes*[7] *Java*. É importante destacar que um arquivo compilado por um compilador *Java* sempre será o mesmo independente do sistema computacional a partir do qual tenha sido gerado. Na fig. 14 é apresentado este fluxo de processamento.

FIGURA 14 - COMPILAÇÃO DE UM PROGRAMA EM *Java*



JVM é o acrônimo para *Java Virtual Machine*, que é o alicerce da linguagem de programação *Java*. *JVM* é uma máquina computacional abstrata formada por um interpretador de *bytecodes Java* e do próprio processador do sistema computacional em que está rodando (fig. 15). Tendo por base um arquivo com *bytecodes*, a *JVM* usa o interpretador *Java* (chamado de "java.exe") para interpretar os *bytecodes* e passar as instruções para o processador. Como os sistemas computacionais podem ser diferentes, por exemplo *Unix*, *Macintosh* ou *Windows*, cada um destes tem que ter o seu próprio interpretador *Java*. Assim, quando um sistema computacional tem um interpretador *Java*, diz-se que este sistema é uma *JVM*.

FIGURA 15 - UMA *Java* VIRTUAL MACHINE

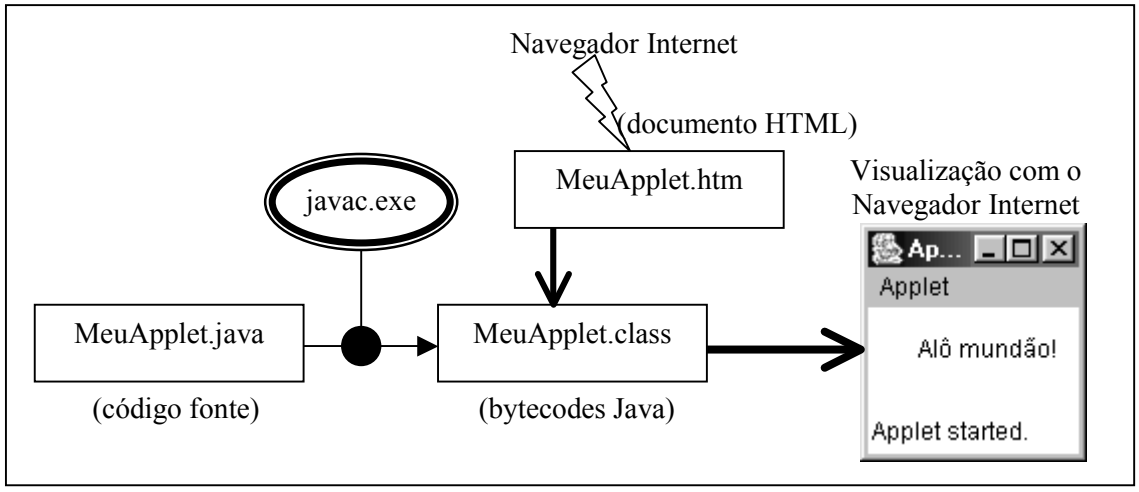


[7] *Bytecode Java* é uma instrução em códigos de bytes para uma *JVM*.

Independentemente de ser uma linguagem de programação e de implementar o conceito de uma *JVM*, diz-se que *Java* é também uma plataforma. Para um programador, uma plataforma é definida pelas *API* (*Application Programming Interfaces*[8]) com as quais o programador conta para escrever os programas (FLANAGAN, 2000, p. 5). Cada sistema operacional tem as suas próprias *API*. Assim, quando se produz um programa tem-se que usar as *API* relativas ao sistema operacional em que este programa será executado. Entretanto, quando se usa a *Java 2 Platform* é possível se escrever aplicativos em *Java* sem se preocupar em duplicá-los com vistas aos diferentes sistemas operacionais em que estes aplicativos irão rodar.

Os programas concebidos com a linguagem *Java* são de dois tipos. Um é chamado de aplicativo *Java* e é executado independente da Web. O outro tipo é chamado de *applet Java*. Este é um programa concebido para rodar na Web e é chamado a partir de um documento HTML. Sua função é criar alguma funcionalidade extra ou interação com o usuário. A idéia é que com um navegador Internet o arquivo HTML é acessado e a partir deste é feita uma chamada ao programa *applet Java* que então realiza alguma operação. Na fig. 16 é apresentado o fluxo de processamento que é iniciado com um navegador Internet e termina com a apresentação em uma janela de um texto simples.

FIGURA 16 - EXEMPLO DE UM PROGRAMA *Applet Java*



[8] API - é um conjunto predefinido de classes *Java* que existe em cada instalação *Java*. Um conjunto de classes afins é agrupado como um pacote. Os pacotes em *Java* são definidos por suas funcionalidades tais como: entrada/saída, rede, gráficos, criação de interface do usuário, segurança e outros.

A versão *Java* mais atual é a *J2SE*, que é o acrônimo para *Java 2 Platform Standard Edition*. Esta versão pode ser obtida livre de taxa no endereço http://java.sun.com/j2se/1.4.2/download.html (acesso em 10 set. 2003). *J2SE* contém J2SDK[9] (*Java 2 Software Development Kit*) e o *JRE*[10] (*Java Runtime Environment*) e as *API* que os desenvolvedores precisam para escrever e rodar seus aplicativos e *applet Java*.

2.7 PADRÃO VRML E O CONJUNTO DE CLASSES GeoVRML

VRML é o acrônimo para *Virtual Reality Modeling Language*. Hoje, VRML é um padrão internacional para disponibilizar cenas e objetos tridimensionais através da Internet. Com um simples arquivo texto, é possível se descrever objetos tridimensionais segundo um cenário e permitir que um observador interaja com os objetos e navegue através da cena usando um visualizador Web3D (ver item 4.4 SISTEMA CORTONA).

Durante a 1ª Conferência da WWW (*World Wide Web*), em 1994, foi apresentado um protótipo de interface tridimensional para a Web chamado *Labyrinth*. Isto criou entre os participantes um consenso sobre a necessidade de uma linguagem comum para descrição de cenas 3D na Web. No início da sua fundação a Web3D Consortium se concentrava apenas nas tecnologias de realidade virtual usadas para a Internet, por isto se chamava VRML Consortium. Em 1999, seu nome foi modificado para Web3D Consortium e o seu enfoque é estendido para incluir todas as tecnologias de Web tridimensionais (Web3D Consortium, 2004).

Como uma atividade do Web3D, em maio de 1995 foi concluída a especificação da VRML 1.0. A restrição que existia para esta versão é que era possível somente se criar cenas estáticas. Neste mesmo ano, teve início um trabalho do VAG (*VRML Architeture Group*), que é formado por colaboradores da comunidade VRML, para incorporar animação e interação à VRML. Posteriormente, algumas empresas e

---

[9] O *J2SDK* (*Java 2 Software Development Kit*) anteriormente denominado de JDK (*Java Development Kit*) é a coleção de programas necessários para escrever aplicativos e *applets Java*.

[10] *JRE* contém o conjunto de programas que são necessários para executar os aplicativos e *applets* em *Java*.

colaboradores cunharam o termo Mundos em Movimento (*Moving Worlds***)** que foi o ponto de partida para a VRML 2.0 (Web3D Consortium, 2004).

Em 1997, teve início um esforço para apresentar a especificação à ISO (*International Standards Organization*) e à IEC (*International Electrotechnical Commission*). Após ser reescrita, a VRML 2.0 foi formalmente liberada como padrão internacional ISO/IEC 14772:1997. Esta versão teve sua sintaxe redefinida e foram acrescentadas modificações para permitir animação, interação, som, efeitos ambientais especiais e extensões à linguagem. Esta é a versão atual, mas é conhecida informalmente pela designação VRML97.

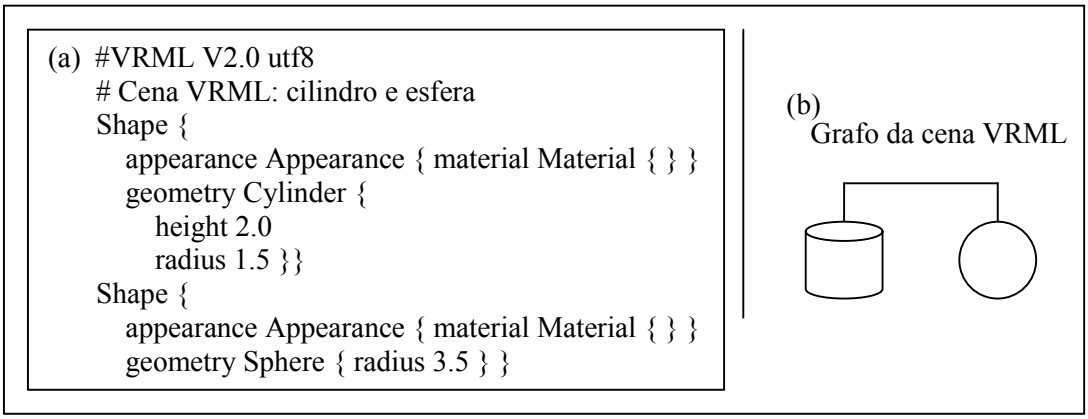
2.7.1 Nó, Campo e Valor de Campo

VRML segue o paradigma de programação orientada a objeto, sendo possível, portanto, criar mundos dirigidos por eventos e compostos por objetos que podem apresentar algum tipo de comportamento. Estes objetos interagem e mantêm relações dinâmicas por meio de mensagens. Os arquivos VRML descrevem mundos e objetos tridimensionais usando grafo de cena hierárquico (CAREY; BELL, 2004). Cada nó do grafo é considerado como um objeto VRML. Para descrever o nó são utilizados campos, que podem assumir valores específicos dentro de um certo domínio, por exemplo, inteiros, reais (precisão simples) e caracteres. Os campos e os seus valores especificam os atributos do nó. Os nós podem conter outros nós, que são chamados de nós filhos enquanto os primeiros são os nós pais.

Os arquivos VRML usam uma extensão "wrl" e podem estar armazenados no disco do computador do usuário ou então serem acessados remotamente via Internet. No que se refere aos tipos de nós existentes em VRML, tem-se o *Shape***,** que descreve as primitivas geométricas básicas que podem ser: *Cone, Cylinder, Sphere* ou *Box.* Na fig. 17a é apresentado um exemplo de arquivo VRML, em que são descritas as primitivas geométricas do tipo cilindro e esfera. Na primeira linha do arquivo é informado ao navegador VRML que este arquivo é um documento VRML de acordo com a sintaxe da versão 2.0. A sigla utf8 identifica um padrão internacional de conjunto de caracteres cujo significado é *Universal Character Set Transformation Format*, de 8-*bit*, que representa um conjunto com mais de 24.000 caracteres para

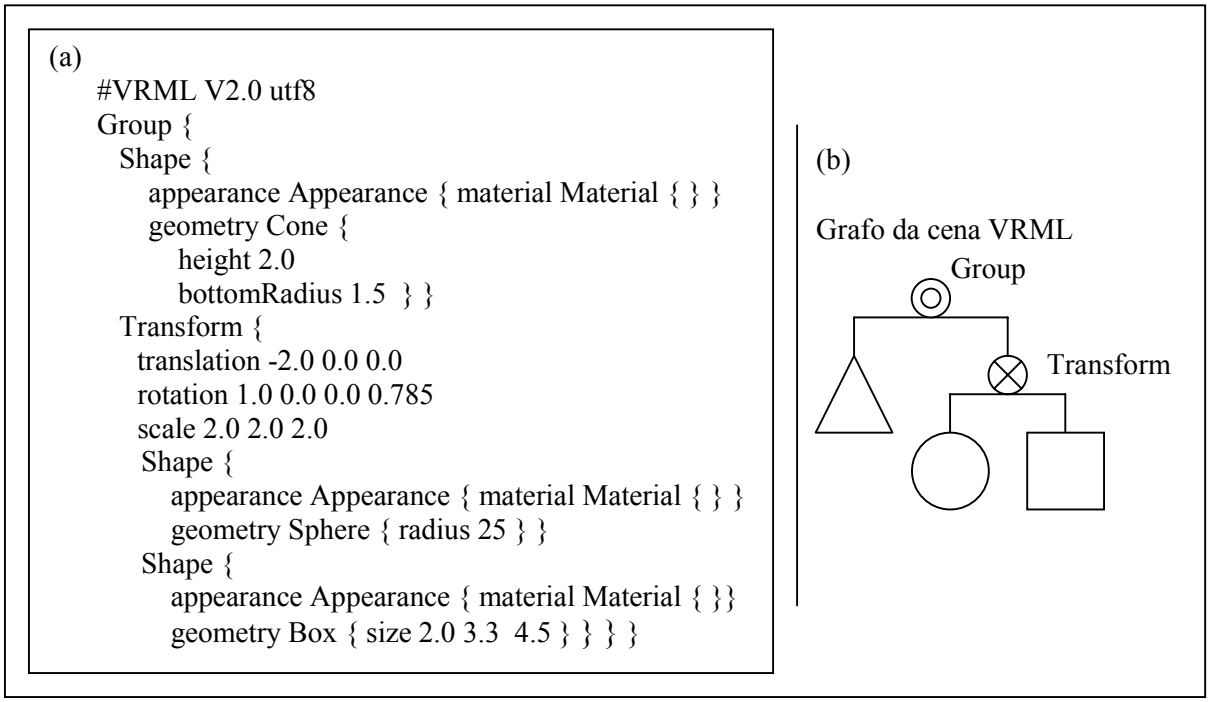
várias línguas. Na fig. 17b é apresentada a estrutura de grafo hierárquico que é formado por dois nós.

FIGURA 17 - EXEMPLO DO CONTEÚDO DE UM ARQUIVO VRML E O RESPECTIVO GRAFO USANDO UM NÓ *Shape*



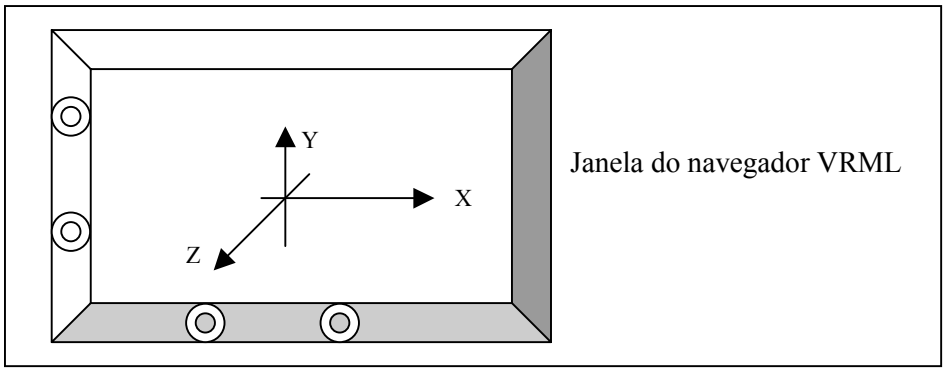
Na fig. 18a é apresentado um outro exemplo em que é utilizado um nó *Group***,** que permite um agrupamento de nós básicos que não necessitam de transformações geométricas. Além disto, neste mesmo exemplo foi usado o nó *Transform,* que também é um nó de agrupamento, mas que permite a aplicação de transformações geométricas (translação, rotação e escala) sobre os objetos VRML. Na fig. 18b é apresentado o grafo hierárquico correspondente.

FIGURA 18 - EXEMPLO DO CONTEÚDO DE UM ARQUIVO VRML E O RESPECTIVO GRAFO USANDO UM NÓ *Group* E UM NÓ *Transform*



Para posicionar os objetos na cena VRML é usado um sistema de coordenadas cartesianas tridimensionais local (x,y,z). Este sistema tem origem no centro da janela do navegador, com os eixos orientados conforme mostrado na fig. 19. A unidade utilizada para estas coordenadas é o metro.

FIGURA 19 - ORIGEM DO SISTEMA DE COORDENADAS SOBRE A JANELA DO NAVEGADOR VRML

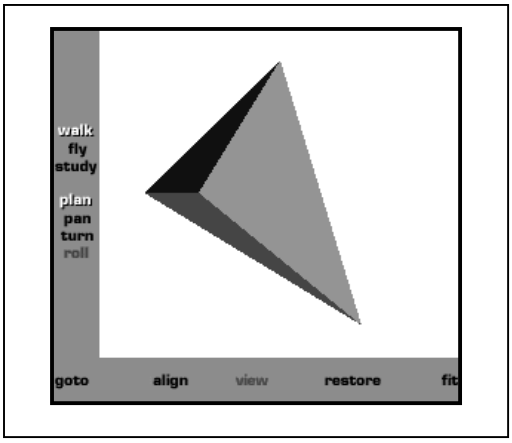


Outros tipos de nós básicos são: *PointSet, LineSet* e *FaceSet*. Com estes pode-se criar, respectivamente, os elementos pontuais, lineares e faces planas. Entretanto, quando se deseja criar um objeto que seja formado por um conjunto de faces planas, utiliza-se o nó *IndexedFaceSet*. Na fig. 20 é apresentado o conteúdo do arquivo VRML com a representação de um tetraedro e na fig. 21 a imagem VRML resultante.

FIGURA 20 - CONTEÚDO DO ARQUIVO VRML USANDO UM NÓ *IndexedFaceSet*

```
#VRML V2.0 utf8
Group {
   children [
   Viewpoint { position  5.  5.  15. }
   Shape { # Shape 1
      appearance Appearance { material Material { } }
      geometry IndexedFaceSet {
         solid FALSE
         coord Coordinate {
            point [  5. 10. 0. ,  0.  5. 0.,  8.  0. 0. ,  2.5 5. 2.5  ]}
            coordIndex [ 0 1 2 -1 , 0 1 3 –1, 1 2 3 -1 , 2 0 3 -1 ]
            color Color { color[ 1 1 0,  0 0 1,  0 1 0,  0 1 1] }
         colorPerVertex FALSE
         colorIndex[0 1 2 3] } } ] }
Background { skyColor 1 1 1 }
```

FIGURA 21 - OBJETO VRML FORMADO POR FACES CRIADAS COM O NÓ *IndexedFaceSet*

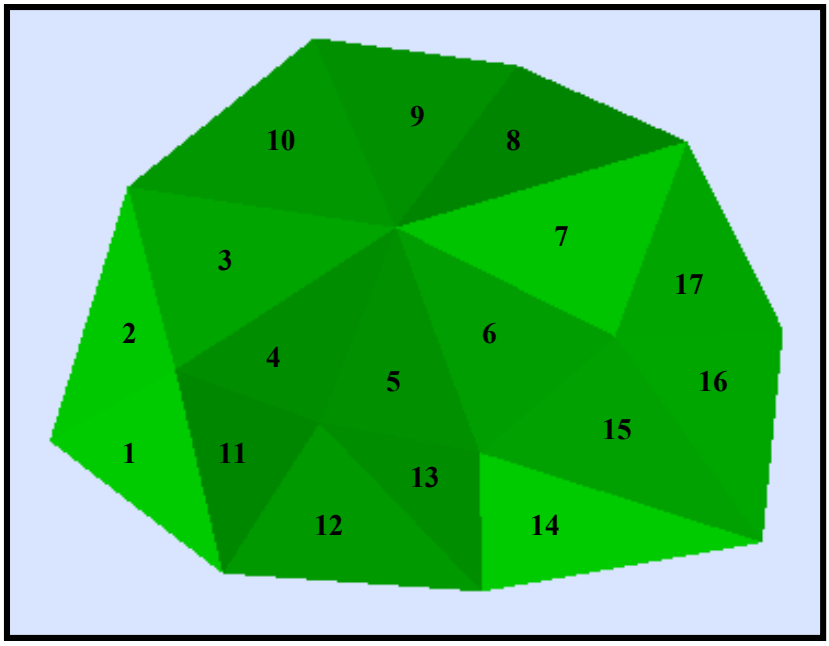


Quando se deseja representar uma superfície irregular, como por exemplo, a superfície topográfica, então além de se poder utilizar o nó *IndexedFaceSet*, tem-se também o nó *ElevationGrid,* que são equivalentes respectivamente aos modelos TIN (*Triangulated irregular Networks*) e DEM (*Digital Elevation Models*) tradicionalmente utilizados para representar a superfície topográfica. Na fig. 22 é apresentado o conteúdo do arquivo VRML que representa uma simulação de uma pequena porção de superfície formada por um conjunto de faces planas triangulares. Na fig. 23 é apresentada a imagem VRML resultante.

FIGURA 22 - ARQUIVO VRML COM SUPERFÍCIE REPRESENTADA POR CONJUNTO DE FACES TRIANGULARES

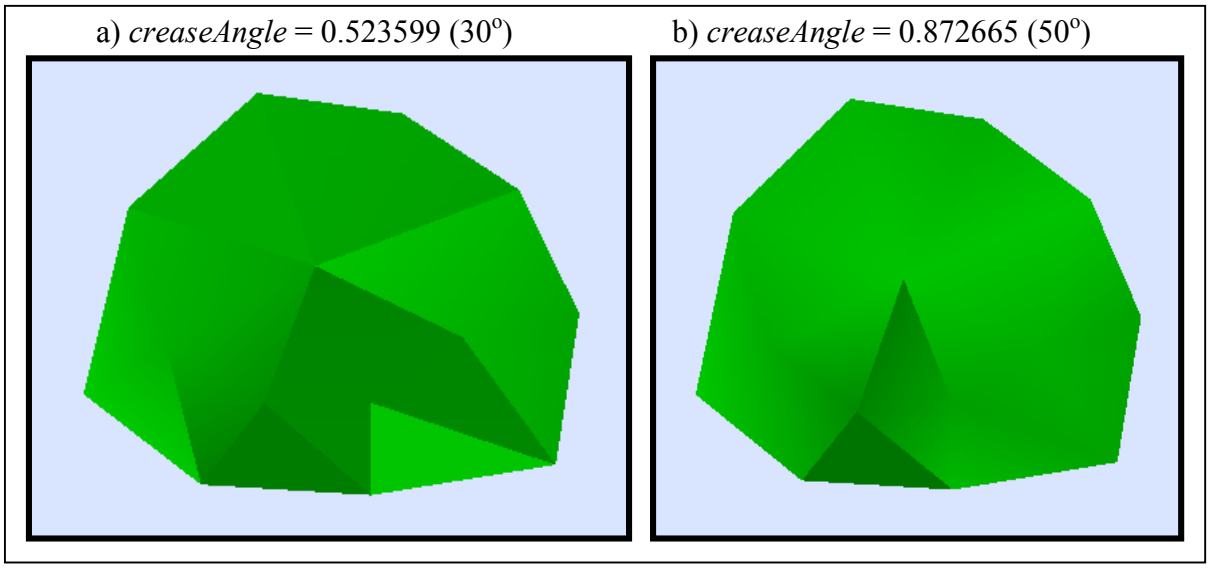
```
#VRML V2.0 utf8
Group {  children [  Viewpoint {
    position  350.   -150.   88.42
    orientation   1  0   0  1.57 }
    Shape {
     appearance Appearance {
      material Material { diffuseColor 0  0.8  0 } }
      geometry IndexedFaceSet {
       coord Coordinate { point[
         298.25   778.31    10.00,   138.76   647.53     1.01,   365.23   590.11  160.00,
         590.11   666.67  100.20,   473.68   755.98     1.50,   185.01   486.44   20.70,
         437.00   416.27   15.56,   527.91   511.96  120.67,   696.97   521.53    1.00,
         668.26   341.31   20.30,   224.88   314.19   13.30,    79.74   425.84   12.20,
         311.00   448.17   150.36,  440.19   296.65     1.05  ] }
       coordIndex[   # FACES
            5  11  10  -1, 12  10  13  -1,  5  10  12  -1,   1  5  2  -1,   5  1  11  -1,  0  2  4  -1,
              2  0  1  -1,    5  12  2  -1,   12  6  2  -1,  6  13  9  -1,  13  6  12  -1,  6  9  7  -1,
              2  3  4  -1,     2  7  3  -1,    8  3  7  -1,   7  9  8  -1,    7  2  6  -1 ]
       solid FALSE  }  }  ]  }
```

FIGURA 23 - SUPERFÍCIE GERADA COMO UM NÓ *IndexedFaceSet*



O nó *IndexedFaceSet* tem um campo que quando inicializado apropriadamente pode produzir uma imagem da superfície suavizada. Este campo denomina-se *creaseAngle* e assume valores angulares em radianos. Se o ângulo entre duas faces adjacentes é menor do que o valor atribuído ao membro *creaseAngle* então as normais a cada face são recalculadas de modo que seja produzido uma suavização na passagem de uma face para outra (CAREY e BELL, 2004). Na fig. 24a é apresentado o efeito resultante quando se utilizou o valor de 0.523599 radianos (30^o) para *creaseAngle*, enquanto que, na fig. 24b é apresentado o efeito produzido para um valor de 0.872665 radianos (50^o). Nestes casos são produzidas suavizações nas regiões de passagem de uma face para outra toda vez que as faces adjacentes têm, respectivamente, ângulos menores do que 30^o e 50^o.

FIGURA 24 - SUAVIAZAÇÃO DAS FACES ADJACENTES COM *creaseAngle*



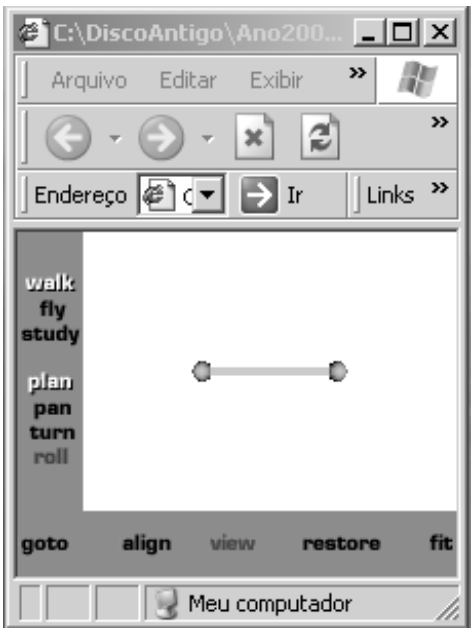
2.7.2 Referência Múltipla

Existem situações em que um mesmo nó é referenciado várias vezes num arquivo VRML. Então, ao invés de se repetir todo o trecho de código relativo a este nó, usam-se as palavras chaves *DEF* e *USE*. Com a palavra chave *DEF* se define um nome para o correspondente nó e se cria este nó. Com a palavra chave *USE* se indica ao *browser* que uma referência ao nó criado deve ser inserida no grafo de cena. Isto permite que se compartilhe um único nó em mais do que uma localização na cena, com a vantagem de se modificar todas essas referências caso o nó venha a ser modificado (fig. 25).

FIGURA 25 - EXEMPLO USANDO AS PALAVRAS CHAVE *DEF* E *USE*

```
#VRML V2.0 utf8
Group {
  children [
    Shape {
      appearance Appearance { material Material { } }
      geometry Box {
        size 4. .2 .2  }  }
    Transform {
      translation -2.0 0.0 0.0
      children DEF esfera Shape {
        appearance Appearance { material Material { } }
        geometry Sphere { radius 0.3 }  }  }
    Transform {
      translation 2.0 0.0 0.0
      children USE esfera  }  ] }
```

b) imagem resultante



2.7.3 Prototipação

Prototipação é o mecanismo que permite que se estendam os tipos de nós existentes. Quando se tem a repetição do mesmo objeto VRML numa cena é possível se utilizar as palavras chaves *DEF* e *USE*. Entretanto, se os objetos apresentam pequenas diferenças nos valores de seus campos, por exemplo, seja um nó do tipo *Sphere* com distintos valores de raio deve-se usar um protótipo e passar um valor do raio toda vez que houver uma instanciação do correspondente nó. Para definir um protótipo usa-se a palavra chave *PROTO*. Adota-se um nome para o novo nó e se declara este nó, definindo os tipos de campos (*field* e *exposedField*), eventos (*eventIns* e *eventOuts*) e valores *default*.

Um exemplo de um protótipo é apresentado na fig. 26. O campo entre colchetes no início da declaração é a interface para o objeto e o número “0.2” é o valor *default* para o campo definido como “raio”. Quando uma instância de “minhaEsfera” é declarada, o valor assumido para o campo “raio” é passado para o campo “radius” do nó *Sphere***,** usando-se para isto a palavra chave *IS*. Quando for desejado declarar um nó “minhaEsfera” com um raio diferente de “0.2”, tem-se que declarar este novo valor

quando declarar o nó “minhaEsfera”, como a seguir:

a) Linha de código: minhaEsfera { }
b) Linha de código: minhaEsfera { raio 0.5 }

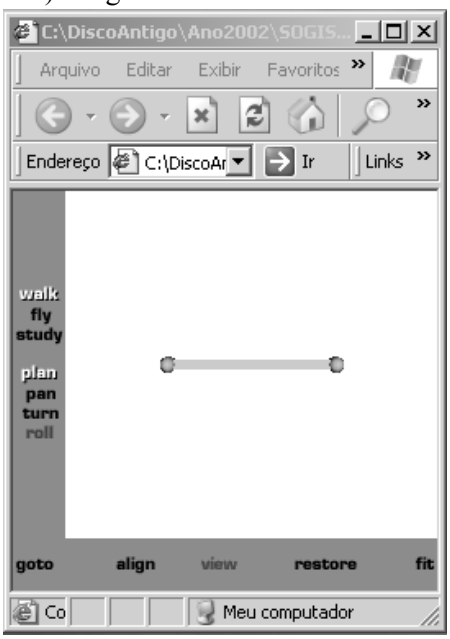
No caso (a) é gerada uma esfera com raio “0.2”, que é o valor *default*. No caso (b) é gerada uma esfera com raio “0.5”, que é o valor passado para o campo “radius”.

FIGURA 26 - EXEMPLO USANDO PROTÓTIPO DECLARADO INTERNAMENTE

a) Conteúdo do arquivo VRML

```
#VRML V2.0 utf8
# Prototipação
PROTO minhaEsfera [
  field SFFloat raio  0.2
] {
 Shape {
   appearance Appearance { material Material { } }
   geometry Sphere { radius IS raio }  } }
Group {
  children [
    Shape {
      appearance Appearance { material Material { } }
}
      geometry Box {
        size 4. .2 .2
      }  }
    Transform {
      translation -2.0 0.0 0.0
      children DEF esfera minhaEsfera { }
    }
    Transform {
       translation 2.0 0.0 0.0
       children USE esfera     }  ]  }
```

b) imagem resultante



É possível também definir *eventIns*, *eventOuts* e *exposedFields* para os protótipos usando a palavra chave *IS***,** que associa um campo do protótipo a um campo do nó existente. Deve ser destacado que um campo do protótipo tem de ser associado com um mesmo tipo de campo ou evento e que estes têm de ter um mesmo tipo de dado.

Existem situações em que se pode desejar definir um protótipo fora do arquivo VRML. Para isto declara-se no arquivo VRML o novo protótipo e informa-se ao *browser* para que procure em um arquivo específico, também declarado, que

contém a definição do novo protótipo (SCHNEIDER, 2004). Para declarar o arquivo externo usa-se a palavra chave *EXTERNPROTO***,** como é mostrado na fig. 26. O arquivo em que está o protótipo deve conter na primeira linha o cabeçalho VRML e então a definição do protótipo, como apresentado na fig. 27.

FIGURA 27 - EXEMPLO USANDO PROTÓTIPO DECLARADO EXTERNAMENTE

a) conteúdo do arquivo VRML

```
#VRML V2.0 utf8
EXTERNPROTO minhaEsfera [
  field SFFloat raio
] "protoEsfera.wrl"
Group {
  children [
    Shape {
      appearance Appearance { material Material { } }
      geometry Box {
        size 4. .2 .2
      } }
    Transform {
      translation -2.0 0.0 0.0
      children minhaEsfera { raio .25 }
    }
    Transform {
       translation 2.0 0.0 0.0
       children minhaEsfera { raio .5 }
    }  ] }
```

b) imagem resultante

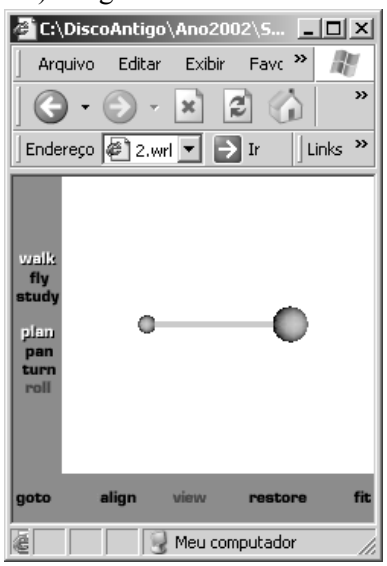


FIGURA 28 - CONTEÚDO DO ARQUIVO "PROTOESFERA.WRL", QUE DEFINE O NOVO PROTÓTIPO

```
#VRML V2.0 utf8
PROTO minhaEsfera [
  field SFFloat raio 0.2
] {
 Shape {
   appearance Appearance { material Material { } }
   geometry Sphere { radius IS raio }
 } }
```

2.7.4 Tempo em VRML

Com VRML é possível se introduzir a noção de tempo e com esta as mudanças que os objetos de uma cena podem sofrer. Tais mudanças podem ocorrer:

a) quando se modifica o estado de um dispositivo de entrada;

b) com o passar do tempo;

c) quando existe movimento do observador ou dos objetos na cena.

Para detectar tais mudanças são utilizados os nós do tipo sensor. De acordo com CAREY e BELL (2004) existem dois grupos de nós sensores, que são chamados sensores ambientais e sensores de dispositivos de apontamento (*pointing-device sensors)*. Os sensores ambientais podem ser: *ProximitySensor; TimeSensor; VisibilitySensor;* e *Collision*. O sensor *ProximitySensor* é usado para detectar quando o usuário navega sobre uma região específica da cena VRML. O sensor *TimeSensor* é um contador de tempo e é usado para iniciar e para parar aqueles nós que estão baseados em tempo (tais como os nós interpoladores). O sensor *VisibilitySensor* detecta quando uma parte específica do mundo torna-se visível para o usuário. O nó *Collision* detecta quando o observador colide contra objetos na cena. Os sensores de dispositivos de apontamento são utilizados para detectar eventos que são provocados pelo usuário quando este usa, por exemplo, o *mouse*. Estes tipos de nós podem ser: *CylinderSensor; PlaneSensor; SphereSensor; TouchSensor;* e *Anchor*. De acordo com CAREY e BELL (2004) um sensor deste tipo é ativado quando o usuário posiciona o dispositivo de apontamento sobre um objeto cuja geometria tem um sensor específico associado.

Numa cena VRML, o *browser* controla a passagem de tempo fazendo com que o sensor de tempo (*TimeSensor*) gere eventos à medida que o tempo passa. A partir dos eventos gerados, os objetos VRML podem ser modificados. Os eventos de tempo que são gerados têm uma marca de tempo *t* (*timestamp*) que caracteriza um instante na escala de tempo. A unidade de tempo usada é o segundo. O instante de tempo zero (0.0) em VRML é equivalente a 00:00:00 GMT (*Greenwich Mean Time*) para 01 de Janeiro de 1970. Conseqüentemente, quando se têm tempos negativos estes

são interpretados como anteriores a 01 de Janeiro de 1970. Processar um evento com uma marca de tempo *t* pode somente resultar na geração de eventos com marcas de tempo maiores ou iguais a *t,* o que equivale dizer que o tempo é contínuo e sempre vai em direção ao futuro.

2.7.4.1 Nó *TimeSensor*

O nó *TimeSensor* é o mais importante dentre os nós sensores, porque este é capaz de detectar passagens de tempo e então enviar este valor para outros nós (LEMAY, COUCH E MURDOCK, 2004). O nó *TimeSensor* é utilizado com os seguintes propósitos:

a) controlar animações;
b) controlar atividades periodicamente;
c) inicializar eventos de ocorrência única, tal como disparar um alarme de relógio.

Na fig. 29 são apresentados os tipos de campo do nó *TimeSensor*. Os campos do tipo *exposedFields* são processados e seus correspondentes eventos (eventOuts) são enviados para outros nós. Quando um nó *TimeSensor* torna-se ativo, ele gera um valor "TRUE" para o campo *isActive* e é iniciada a geração de eventos do tipo *time, fraction_changed* e *cycleTime*. O evento *time* envia o tempo absoluto para um dado instante. O evento *fraction_changed* produz valores reais no intervalo fechado [0, 1]. O evento *cycleTime* envia um evento de tempo para o *startTime* e para o início de cada novo ciclo. Para o *startTime* (instante inicial) o valor de *fraction_changed* é 0.

FIGURA 29 - OS CAMPOS DO NÓ *TimeSensor*

```
TimeSensor {
  exposedField  SFTime   cycleInterval   1       # (0,∞)
  exposedField  SFBool   enabled        TRUE
  exposedField  SFBool   loop           FALSE
  exposedField  SFTime   startTime      0       # (-∞,∞)
  exposedField  SFTime   stopTime       0       # (-∞,∞)
  eventOut      SFBool   IsActive
  eventOut      SFTime   time
  eventOut      SFFloat  fraction_changed
  eventOut      SFTime   cycleTime }
```

2.7.4.2 Nós do tipo interpolador

De acordo com LEMAY, COUCH e MURDOCK (2004) quando ocorrem mudanças nos objetos de uma cena VRML os nós interpoladores representam o elemento de ligação entre os nós no grafo de cena, porque eles são responsáveis por traduzir a noção de passagem de tempo em valores que serão utilizados para provocar as mudanças nos objetos. Existem seis tipos de interpoladores, que são: *CoordinateInterpolator; OrientationInterpolator; NormalInterpolator; ColorInterpolator; PositionInterpolator;* e *ScalarInterpolator*. Todos têm os mesmos campos (fig. 30), mas com tipos diferentes para *keyValue* e *value_changed*.

FIGURA 30 - CAMPOS PARA DOIS TIPOS DE NÓ INTERPOLADOR

```
CoordinateInterpolator {
  eventIn       SFFloat    set_fraction
  exposedField  MFFloat    key          [ ]
  exposedField  MFVec3f    keyValue     [ ]
  eventOut      MFVec3f    value_changed
}
```

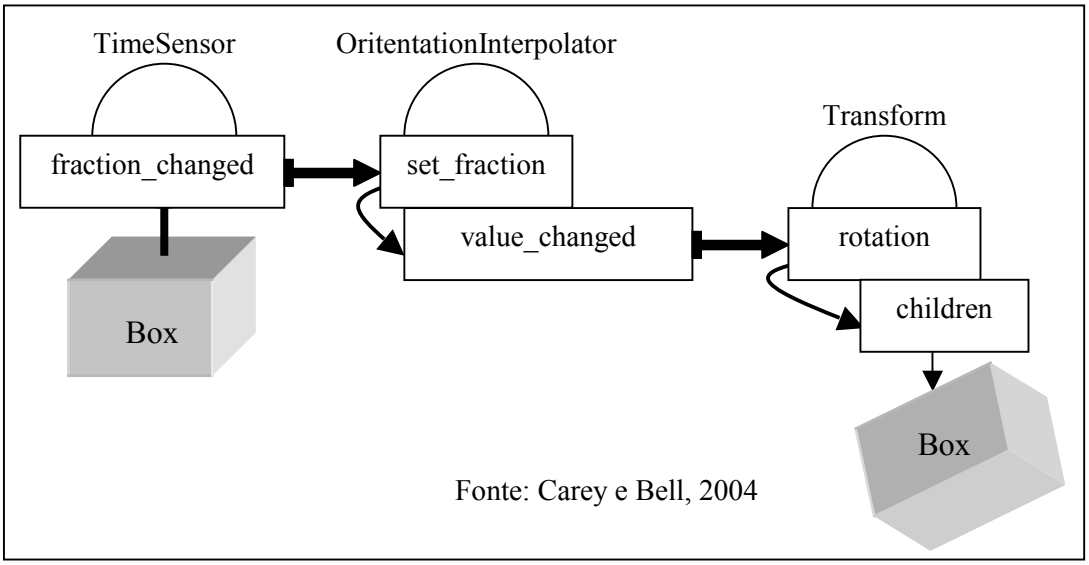
```
OrientationInterpolator {
  eventIn       SFFloat    set_fraction
  exposedField  MFFloat    key          [ ]
  exposedField  MFRotation keyValue     [ ]
  eventOut      SFRotation value_changed
}
```

TABELA 1 - TIPOS DE NÓS INTERPOLADORES

Nó	Aplicação
CoordinateInterpolator	interpolar valores de coordenadas
NormalIntepolator	interpolar valores de normais
OrientationInterpolator	interpolar valores angulares que são usados para fazer rotações visando obter uma certa orientação
PositionInterpolator	interpolar valores de posição 3D
ScalarInterpolator	interpolar valores de reais

Associado com o objeto (ou grupo de objetos) que pode apresentar modificações com o passar do tempo, tem-se um nó *TimeSensor*. A partir do momento que este nó está ativo (campo *enabled* é "TRUE") ele envia mensagens para o nó interpolador. Então o nó interpolador realiza uma interpolação linear para o conjunto de valores chamados de valores-chave (*keyValues*) com base nos valores *keys*. Em seguida, o nó interpolador envia mensagens para o nó *Transform* para que este aplique as mudanças sobre o objeto da cena (fig. 31).

FIGURA 31 - FLUXO DE MENSAGENS PARA GERAR MUDANÇAS NUM OBJETO



Para realizar a interpolação o nó interpolador usa uma função *f(t)* linear e por partes (*piecewise-linear function*) sobre um intervalo $(-\infty, +\infty)$. Esta função é definida para *n* valores de *t*, chamados de *key*, e os *n* valores correspondentes de *f(t)*, chamados de *keyValues* (SCHNEIDER, 2004). Os valores *keys* devem ser monotonicamente crescentes. O nó interpolador avalia *f(t)* para um dado valor de *t* (por meio do *set_fraction*) como segue: sejam os *n* valores $t_0, t_1, t_2, ..., t_{n-1}$ (*keys*) que particionam o domínio $(-\infty, +\infty)$ em $n+1$ subintervalos dados por $(-\infty, t_0), [t_0, t_1), [t_1, t_2), ... , [t_{n-1}, +\infty)$. A estes estão associados os valores $v_0, v_1, v_2, ..., v_{n-1}$ de *f(t)* (*keyValues*). A função *f(t)* é definida para ser

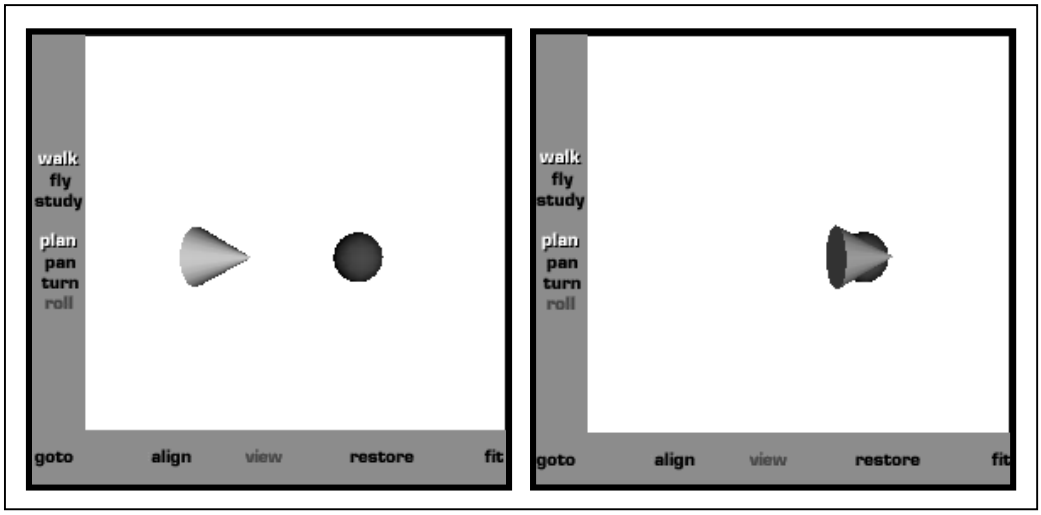
$f(t) = v_0$, if $t <= t_0$,

$= v_{n-1}$, if $t >= t_{n-1}$,

$= \text{linterp}(t, v_i, v_{i+1})$, if $t_i <= t <= t_{i+1}$

em que linterp(t,x,y) é o interpolante linear, i pertence {0,1,.,n-2}. Na fig. 32 são apresentados dois quadros de uma cena VRML que é composta por uma esfera estática e um cone que fica girando sem parar em sua volta.

FIGURA 32 - DOIS QUADROS DE UMA CENA VRML



2.7.4.3 Conjunto de classes GeoVRML

De acordo com RHYNE (2004) existem dois aspectos que restringem o uso da VRML para visualizar dados referenciados à superfície terrestre. O primeiro é que em VRML as coordenadas de uma posição são representadas com precisão simples. Em geral, esta forma de representação não é suficiente devido à ordem de grandeza das coordenadas geográficas. O segundo aspecto restritivo é que VRML usa somente um sistema de coordenadas cartesianas. Com isto, não pode-se visualizar dados geográficos que estejam expressos por coordenadas elipsoidais como latitude e longitude, ou então associados com alguma projeção cartográfica, como por exemplo, a UTM (*Universal Transverse Mercator*) .

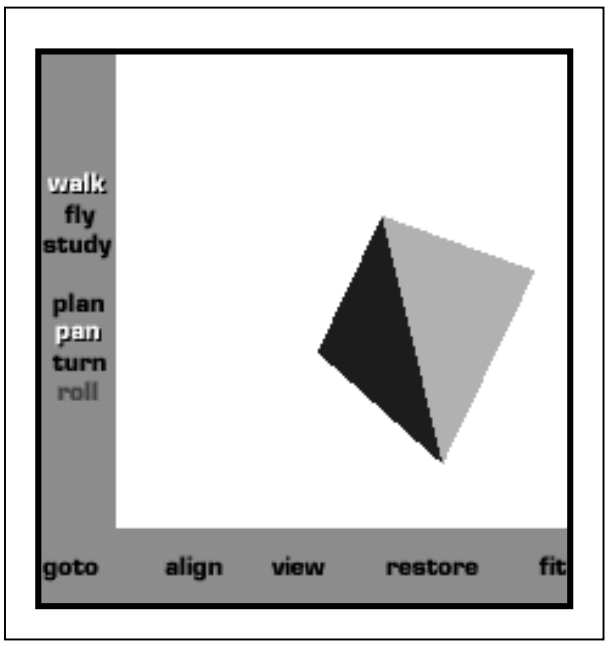
A partir de 1998 foi formado no âmbito do Web3D Consortium o grupo de trabalho GeoVRML. Este grupo discute e estabelece especificações para que se possa usar dados geográficos com VRML. Atualmente existe uma recomendação prática para que se utilize o pacote de classes chamado *GeoTransform* (Web3D Consortium, 2004a). Este pacote é constituído por um conjunto de métodos e classes implementados em linguagem Java que possibilitam lidar com cenas VRML referidas a um sistema de coordenadas geodésicas ou cartográficas. Na tab. 2 são listados alguns nós que foram implementados pelo grupo de trabalho GeoVRML para a integração de dados geográficos à VRML.

TABELA 2 - TIPOS DE NÓS GeoVRML

Nome do nó	Funcionalidade do nó
GeoCoordinate	Constrói a geometria do objeto usando coordenadas geográficas
GeoElevationGrid	Define uma malha regular de pontos usando coordenadas geográficas
GeoInline	Acessa um file com o controle sobre quando carregar e descarregar os dados
GeoLocation	Georreferencia um objeto sobre a superfície terrestre numa cena VRML
GeoLOD	Gerencia os níveis de detalhamento para uma superfície com multi-resolução
GeoMetadata	Inclui um subconjunto genérico de metadados sobre os dados geográficos
GeoOrigin	Especifica um sistema de coordenadas local para aumentar a precisão
GeoPositionInterpolator	Objetos animados dentro de um sistema de coordenadas geográficas
GeoTouchSensor	Retorna a coordenada geográfica do objeto que está sendo apontado
GeoViewpoint	Especifica pontos de vista usando coordenadas geográficas
Fonte: adaptado de http://www.geovrml.org/2.0/doc/nodes.html	

Na fig. 33 é apresentada uma imagem VRML formada por duas faces planas adjacentes, que são definidas espacialmente por suas coordenadas geodésicas (latitude, longitude e altitude). Neste exemplo foram utilizados os nós *GeoCoordinate*, *GeoViewpoint* e *GeoLocation* declarados como protótipos externos.

FIGURA 33 - FACES REPRESENTADAS POR CLASSES GeoVRML USANDO COORDENADAS GEODÉSICAS



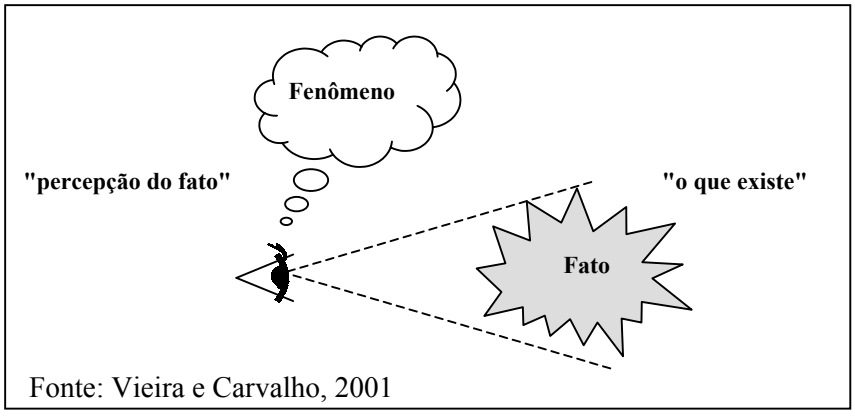
3 MODELO PROPOSTO

O primeiro aspecto a ser considerado quanto ao modelo proposto diz respeito à terminologia utilizada para sua caracterização. Para isto são apresentadas definições e premissas para as quais este modelo é considerado apropriado. O modelo proposto é caracterizado por três classes principais, que são “FenomenoTerrestre”, “AreaLevantamento” e “Recobrimento”. A primeira classe é voltada para a representação de um fenômeno terrestre. A segunda classe representa o conjunto de fenômenos levantados e para este é associado um valor único de escala e de instante de tempo em que o conjunto foi observado. A última classe é usada para representar as variações temporais ocorridas entre os fenômenos.

3.1 FATOS E FENÔMENOS

O USGS (*United States Geological Survey*) apresenta dentro do contexto do SDTS (acrônimo para *Spatial Data Transfer Standard*) a seguinte definição para fenômeno: *"... a fact, occurrence, or circumstance ..."* (USGS, 2004a). Por outro lado, BARROS e LEHFELD (1986, p. 63) reportam que fato quer dizer: "... aquilo que acontece na realidade, o que existe", enquanto fenômeno: "... é a percepção do fato pelo estudioso para um procedimento de análise". Na fig. 34 é apresentado o relacionamento entre os conceitos fato e fenômeno de acordo com esta última definição, ou seja, um fenômeno é uma forma de racionalização sobre um fato que é observado e não o próprio fato.

FIGURA 34 - RELAÇÃO ENTRE FATO E FENÔMENO



Fonte: Vieira e Carvalho, 2001

No contexto desta tese se está utilizando as definições de fato e fenômeno como relatadas por BARROS e LEHFELD (1986, p. 63) porque com esta é possível se contemplar a idéia de que para um mesmo fato podem existir diferentes entendimentos, ou seja, dependendo do ponto de vista do observador, um mesmo fato pode ser percebido de forma distinta. Por exemplo, um fato que ocorre sobre a superfície terrestre pode ser entendido como um fenômeno econômico, político, geográfico, geológico, ou então como puramente topográfico, ou seja, algo que existe sobre a superfície e que é mensurável. Esta premissa está sendo usada para o modelo proposto uma vez que se tem em vista a representação de fenômenos terrestres do ponto de vista da cartografia topográfica.

3.2 CARACTERÍSTICAS DE UM FENÔMENO TERRESTRE

Para caracterizar um fenômeno terrestre dentro do contexto da cartografia topográfica se está admitindo que este fenômeno faz parte de uma única classe de objetos, ou seja, se está considerando que um fenômeno não pode pertencer a mais do que uma classe. Com isto, os fenômenos terrestres de uma classe apresentam o mesmo comportamento e compartilham os mesmos atributos, que são as características associadas com cada semântica.

A semântica de um fenômeno terrestre fica estabelecida por quem faz o seu levantamento. Como conseqüência todos os fenômenos levantados têm uma semântica definida. Por exemplo, um tipo de fenômeno terrestre pode ser um imóvel ou então um tipo de solo. Associado com cada semântica, se imóvel ou solo, tem-se o conjunto de características dos objetos da classe. Para os objetos da classe “Imóvel”, pode-se ter como características: o registro, o nome do proprietário e o tipo de imóvel; para os objetos da classe “Solo”: o tipo de solo e o seu uso.

Além da semântica cada fenômeno terrestre é caracterizado espacialmente pela sua “superfície”, que é a sua parte externa. Entretanto, existem aplicações dentro da cartografia topográfica em que é possível admitir que alguns fenômenos sejam representados com uma dimensionalidade menor: 2D, 1D ou 0D. No caso da dimensionalidade 2D a superfície é representada por uma única face poligonal plana. Nos casos 1D e 0D se tem uma superfície degenerada e representada, respectivamente

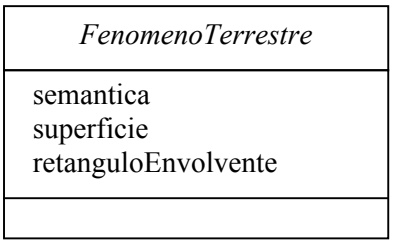
por uma seqüência de pontos (linha) e por um único ponto.

Por exemplo, é possível se adotar a dimensionalidade 1D para representar uma rede de drenagem numa aplicação em que sejam relevantes somente as conexões e caminhos pela rede. Entretanto, em uma outra aplicação, pode ser necessário se ter que representar, além das conexões e caminhos pela rede, as margens dos rios e as suas declividades, dimensionalidade 3D. Um exemplo de dimensionalidade 0D pode ser aquele quando se quer representar a sede administrativa de cada cidade dentro do estado. Neste caso, com apenas um terno de coordenadas e o nome da cidade, pode-se caracterizar espacialmente cada sede administrativa. Deve-se destacar aqui que o conceito de dimensionalidade está relacionado com a idealização da superfície do fenômeno tendo em vista uma certa aplicação.

3.3 REPRESENTAÇÃO DE UM FENÔMENO TERRESTRE SEGUNDO O PADRÃO UML

Como foi considerado no item anterior, todo fenômeno terrestre fica definido por uma única semântica e uma superfície. Adicionalmente, se está usando também um retângulo que envolve esta superfície e permite que se identifique, de imediato, se dois fenômenos são disjuntos. Então a representação de um fenômeno terrestre por meio da UML pode se dar como uma classe abstrata com apenas três membros (fig. 35).

FIGURA 35 - MEMBROS DA CLASSE ABSTRATA FenomenoTerrestre

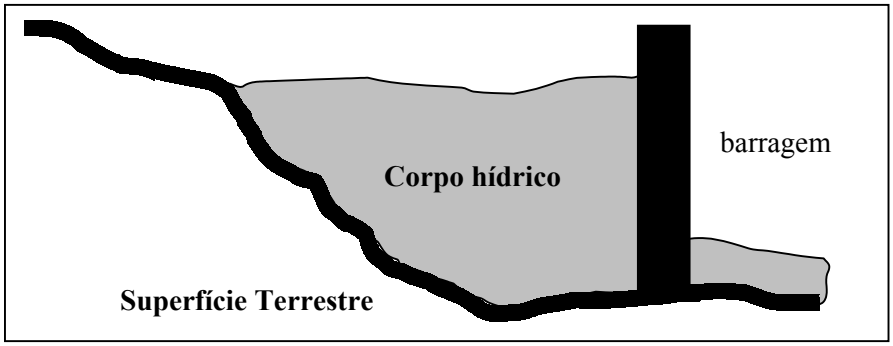


3.4 IDENTIFICAÇÃO DOS FENÔMENOS TERRESTRES

Para identificar os fenômenos terrestres a serem modelados, partiu-se da idéia de que a maior distinção que existe entre estes está na divisão entre regiões de terra firme e de corpos hídricos, que é um ponto de vista comumente aceito na

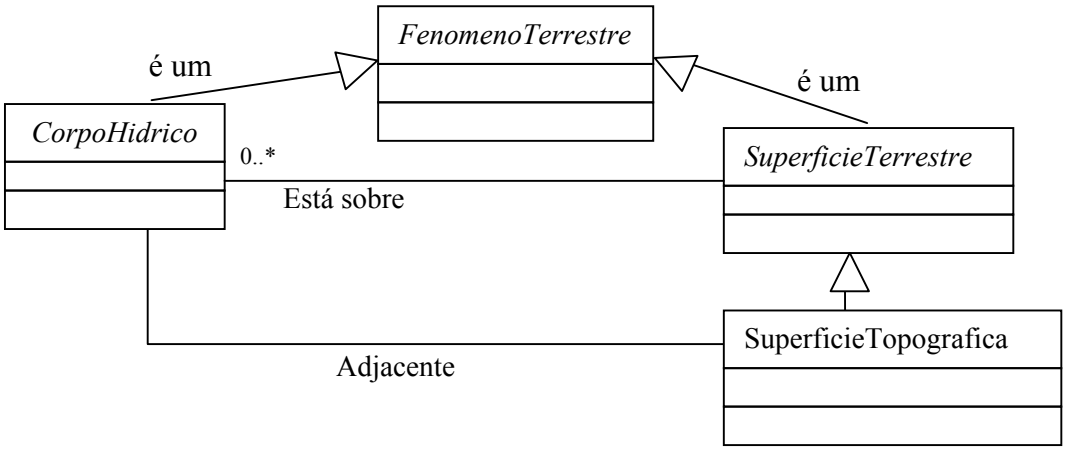
cartografia topográfica (KEATES, 1980, p. 20). Com isto, identificam-se de imediato duas grandes classes de fenômenos terrestres: a superfície terrestre e os corpos hídricos. A Superfície terrestre - é a parte externa da Terra e sobre esta repousam todos os tipos de fenômenos terrestres. Corpo hídrico - é uma massa amórfica em estado líquido que assume a forma do talvegue (ou depressão) em que está contida. Esta massa líquida pode fluir livremente de acordo com o desnível do talvegue, ou então, pode estar represada sobre parte da superfície terrestre. Exemplos de corpos hídricos são os oceanos, os lagos e os rios. O tipo de relacionamento entre os corpos hídricos e a superfície terrestre pode ser enunciado como: “o corpo hídrico está sobre a superfície terrestre" (fig. 36).

FIGURA 36 - VISTA DE PERFIL DE UM CORPO HÍDRICO SOBRE A SUPERFÍCIE TERRESTRE



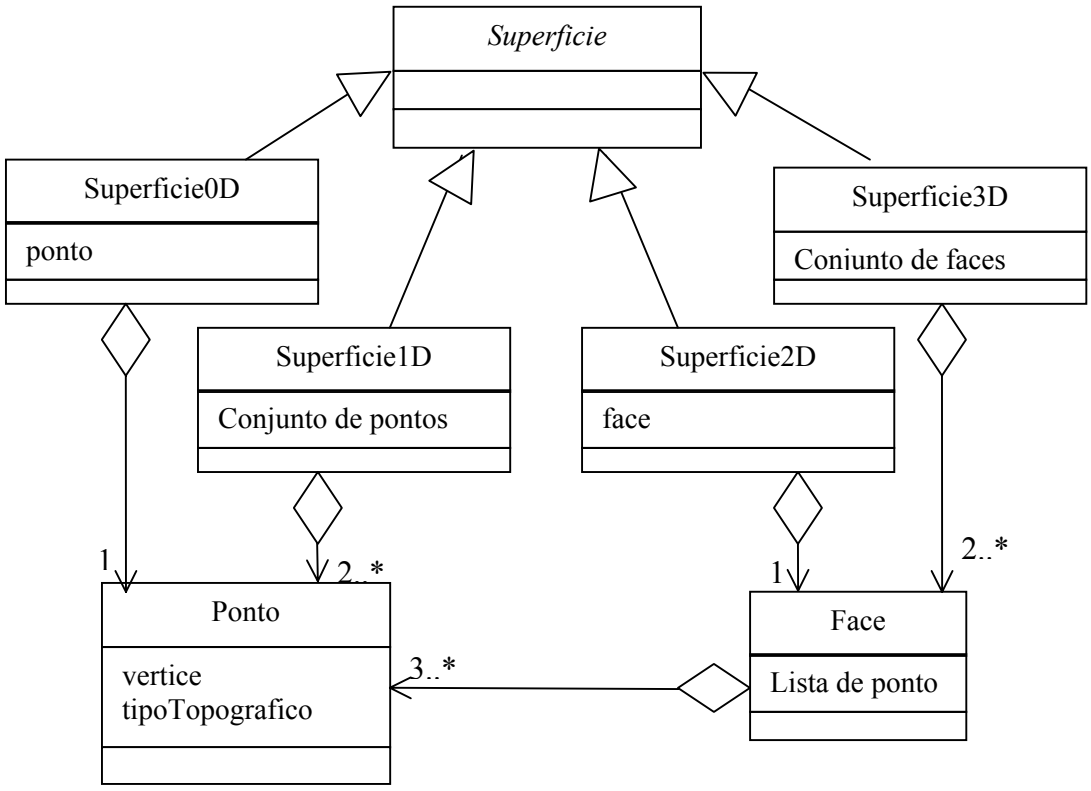
Em conseqüência do relacionamento entre a superfície terrestre e o corpo hídrico, identifica-se um outro fenômeno, que é a superfície topográfica, ou seja, a região da superfície terrestre que não está submersa por qualquer corpo hídrico e, portanto, pode ser visualizada por um observador acima da superfície terrestre. O relacionamento que existe entre a superfície topográfica e um corpo hídrico pode ser enunciado como: “a superfície topográfica e o corpo hídrico são adjacentes". Na fig. 37 são apresentadas estas classes e seus relacionamentos, segundo o padrão UML, destacando-se neste caso que a superfície topográfica é uma especialização da superfície terrestre.

FIGURA 37 - AS CLASSES SUPERFÍCIE TERRESTRE, CORPO HÍDRICO E SUPERFÍCIE TOPOGRÁFICA E SEUS RELACIONAMENTOS



Para modelar a superfície dos fenômenos foi concebida a classe abstrata "Superfície" e as suas correspondentes especializações para representar os fenômenos terrestres com as dimensionalidades 0D, 1D, 2D e 3D. Tais classes foram denominadas: "Superficie0D", "Superficie1D", "Superficie2D" e "Superficie3D". Um fenômeno com valor de dimensionalidade igual a zero tem uma superfície que se degenera num único objeto do tipo "Ponto". Um fenômeno com dimensionalidade igual a um tem a sua superfície degenerada numa seqüência ordenada de objetos do tipo "Ponto". Um fenômeno com dimensionalidade igual a dois tem uma superfície caracterizada por um único objeto do tipo "Face". Um objeto com dimensionalidade igual a três tem uma superfície caracterizada por um conjunto de objetos do tipo "Face" (fig. 38). No contexto deste trabalho se adotou que todas as faces são planas no espaço tridimensional e que as superfícies curvas podem ser particionadas em um conjunto de faces planas. Com este procedimento, teve-se por objetivo apenas simplificar o processo de implementação dos métodos.

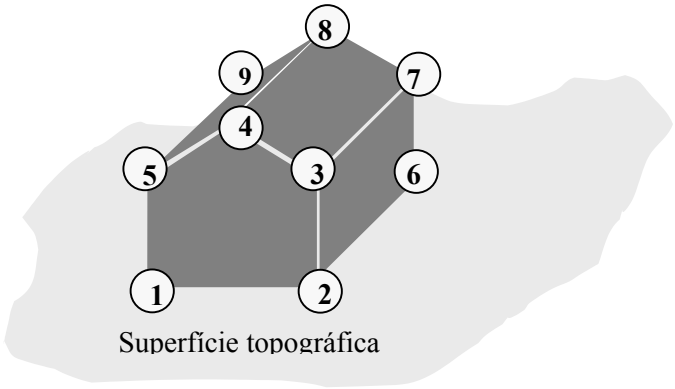
FIGURA 38 - ESPECIALIZAÇÃO DA CLASSE “Superficie”



Para representar uma face se concebeu a classe “Face”, que tem como único membro um conjunto de objetos do tipo “Ponto”. A classe “Ponto”, por sua vez, é a classe mais elementar para representar um fenômeno terrestre. Esta classe tem o membro “Vertice”, que identifica e representa uma localização no espaço em relação a algum sistema geodésico usado como referência. O membro “tipoTopografico” indica se o ponto faz parte ou não da superfície topográfica. Neste trabalho, considera-se que um fenômeno terrestre é constituído por pontos que pertencem à superfície terrestre (ao menos um) e por pontos que formam somente a sua superfície, mas não necessariamente pertencem a superfície terrestre.

Na fig. 39 é apresentado um exemplo em que o fenômeno “Casa” é formado por um conjunto de faces e estas são constituídas por um conjunto de pontos, que estão numerados. Enquanto os pontos 3, 4, 5, 7, 8 e 9 são usados somente para formar as faces da “Casa”, os pontos 1, 2, 6 e 10 além de serem usados para formar as faces da “Casa”, pertencem também a superfície topográfica.

FIGURA 39 - OS PONTOS 1, 2, 6 E 10 SÃO COMUNS A “Casa” E A “SuperfícieTopográfica”



Para se identificar os outros fenômenos terrestres foi realizado um estudo que teve por base o documento "Padronização dos Níveis das Entidades e das Convenções Cartográficas - Escalas Grandes - para uso em Cartografia Digital", proposto pela CTCG (Câmara Técnica de Cartografia e Geoprocessamento[1]), e o documento "Tabelas da Base Cartográfica Digital", proposto pela DSG (Diretoria do Serviço Geográfico do Exército), que descrevem os diferentes tipos de fenômenos representados em cartas topográficas digitais. O estudo realizado consistiu na análise e classificação dos fenômenos de modo a se chegar a um conjunto mínimo, aqui denominado de conjunto essencial de fenômenos. O que se espera com isto é que a partir deste conjunto sejam derivados os outros fenômenos, ou então que da combinação destes sejam formados outros fenômenos. Além dos fenômenos: superfície terrestre, superfície topográfica e corpos hídricos, tem-se:

a) **imóvel** - é toda região delimitada da superfície terrestre, cujo proprietário detém os direitos de usos e frutos sobre ele;
b) **construção -** pode ser de dois tipos (edificação ou artefato de engenharia):

b.1) **edificação** - é toda construção feita pelo homem, de caráter permanente, de um tipo de material, com um certo fim (serviço público, comercial, residencial ou comercial-residencial), delimitada por paredes e teto e

[1] Grupo de trabalho responsável pela apresentação de especificações para a produção cartográfica digital feita e utilizada pelas concessionárias públicas do estado do Paraná.

contida em um imóvel;

b.2) **artefato de engenharia** - é toda construção feita pelo homem, de caráter permanente, mas não edificação, e é utilizada como apoio a alguma atividade humana ou como complementação de uma edificação;

c) **via de acesso** - é toda faixa da superfície terrestre a partir da qual se pode deslocar de um extremo para o outro;

d) **afloramento rochoso** - é todo tipo de rocha que está exposta sobre a superfície terrestre;

e) **solo** - é o extrato superficial que suporta a cobertura vegetal e as árvores;

f) **depósito superficial -** é todo tipo material acumulado na superfície topográfica oriundo de degradação de rochas (exemplo: areia, cascalho etc.);

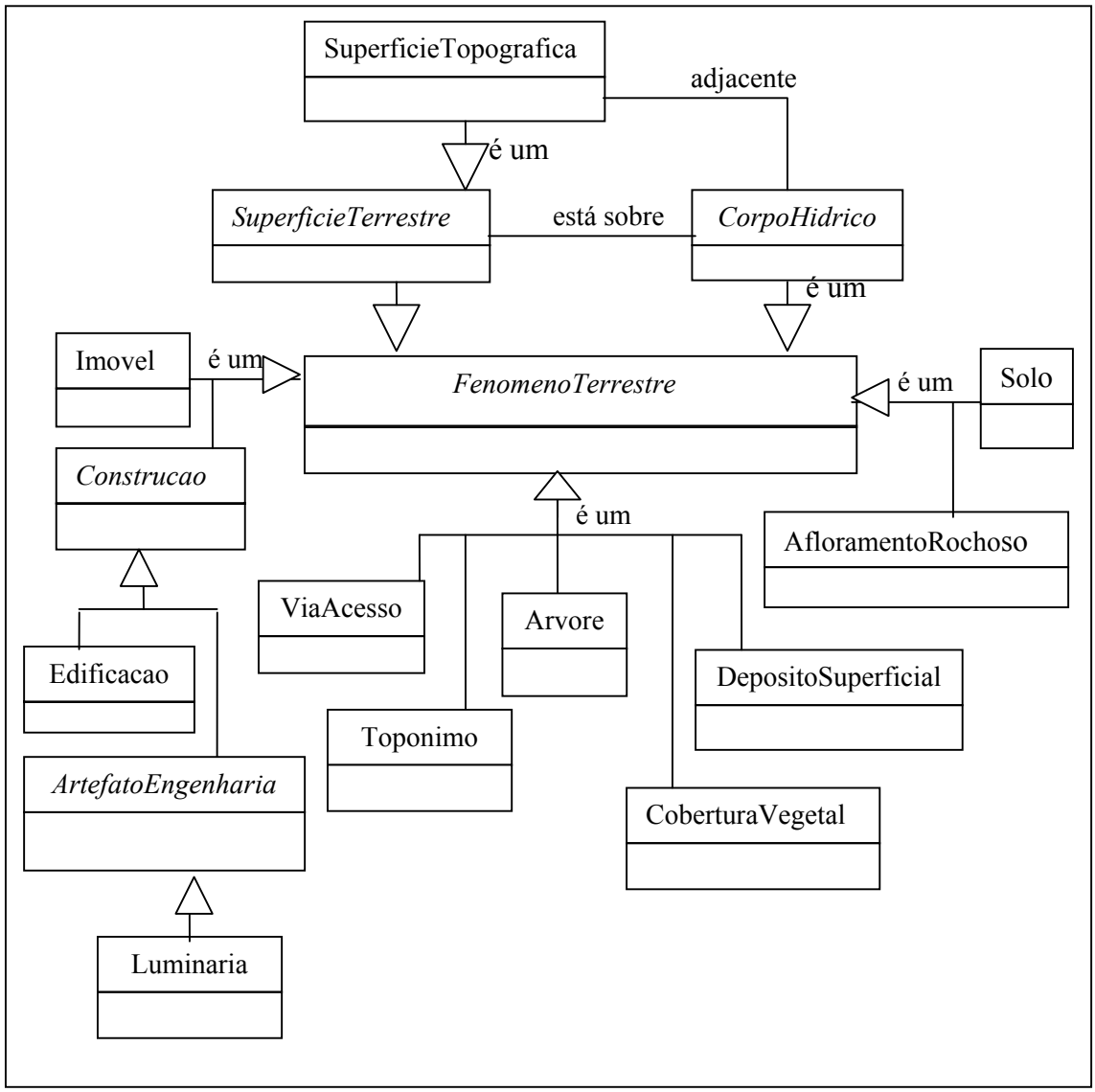
g) **cobertura vegetal** - é toda área de vegetação para a qual se faz uma classificação estruturada. Nesta classificação se leva em conta somente a aparência e a natureza da vegetação predominante dentro da área;

h) **árvore** - é todo tipo de vegetação que é identifica e levantada isoladamente;

i) **topônimo** - é o nome próprio que identifica um fenômeno terrestre.

Na fig. 40 são representadas todas as classes modeladas segundo o padrão UML. A partir da classe “Fenômeno Terrestre” são especializadas diretamente as classes: “Superfície Terrestre”; “Corpo Hídrico”; “Imóvel”; “Construção”; “Via de Acesso”; “Afloramento Rochoso”; “Solo”; “Depósito Superficial”; “Cobertura Vegetal”; ‘Árvore”; e “Topônimo”. Além destas, tem-se também as classes ”Edificação” e “Artefato de Engenharia”, que são especializações da classe “Construção”.

FIGURA 40 - OUTROS FENÔMENOS TERRESTRES



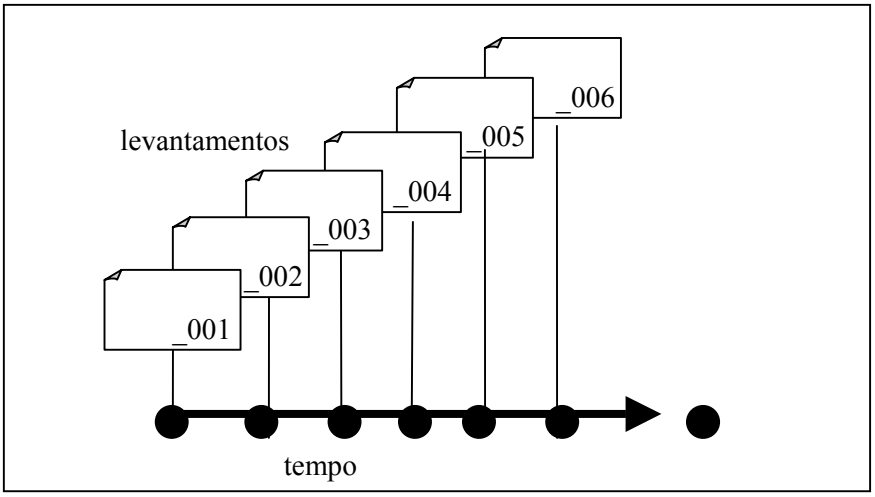
3.5 REPRESENTAÇÃO DA VARIAÇÃO TEMPORAL

Se não fosse necessário representar a variação temporal que um fenômeno terrestre pode apresentar com o passar do tempo, a classe “FenômenoTerrestre” seria praticamente suficiente no processo de modelagem. Entretanto, como se está supondo que pode existir variação temporal para os fenômenos, é necessário representar esta variação no modelo. Para tanto foram concebidas as classes “AreaLevantamento” e “Recobrimento”.

No contexto do modelo proposto, o termo levantamento se refere aos registros de dados que foram obtidos a partir das observações feitas sobre os fatos. Na

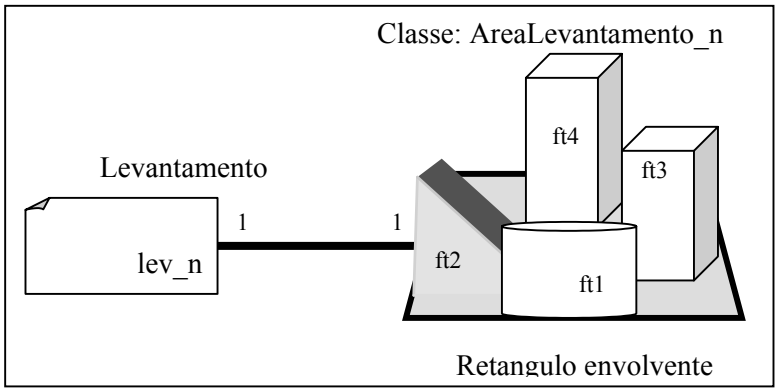
fig. 41 é representado de forma esquemática um conjunto de levantamentos que se supõe que foram realizados com o passar do tempo.

FIGURA 41 - ÁREAS DE LEVANTAMENTO OBTIDAS COM O PASSAR DO TEMPO



Com base nos dados oriundos de levantamentos é possível, então, se fazer a instanciação de cada objeto da classe "AreaLevantamento". Esta classe foi projetada para representar uma área da superfície terrestre que contém um conjunto de fenômenos que foram levantados em um mesmo instante e em uma mesma escala. Conseqüentemente, a escala e o instante de observação estão associados com o levantamento e não com os fenômenos. Esta é uma premissa válida em cartografia topográfica. A vantagem de se adotar valores únicos para a escala e o instante de observação dos fenômenos é que com isto estes membros fazem parte somente da classe "AreaLevantamento" e não necessitam, portanto, serem explicitados em cada fenômeno isoladamente. Como um objeto "AreaLevantamento" é instanciado a partir de um levantamento, tem-se que a relação entre estes é um para um (fig. 42).

FIGURA 42 - A CLASSE AreaLevantamento CONTÉM TODOS OS FENÔMENOS TERRESTRES LEVANTADOS



Dentre os membros da classe "AreaLevantamento" estão: a escala de levantamento; o instante de tempo em que o levantamento foi realizado; o retângulo que envolve toda a área levantada e que é caracterizado por um objeto do tipo "Retangulo". Além destes membros, também se tem os registros de dados do levantamento e o conjunto de fenômenos terrestres levantados (fig. 43).

FIGURA 43 - MEMBROS DA CLASSE "AreaLevantamento" SEGUNDO O PADRÃO UML

AreaLevantamento
Escala InstanteObservação RetanguloEnvolvente RegistroDados ConjuntoFenomenos

Deve ser destacado aqui que, segundo o modelo proposto, está-se classificando os dados como tendo apenas características espaciais e semânticas e que estas podem assumir ou não novos valores com o passar do tempo. Para se determinar se um fenômeno teve alguma variação temporal é necessário comparar os valores dos correspondentes atributos obtidos a partir de levantamentos. Quanto às principais responsabilidades da classe "AreaLevantamento" destacam-se:

a) informar qual a origem do levantamento;

b) informar sobre o retângulo que a envolve a área de levantamento;

c) informar sobre a escala de levantamento;

d) informar sobre o instante de realização do levantamento;

e) informar sobre os fenômenos que estão contidos na área levantada.

3.5.1 Variação Temporal

Para se determinar a variação temporal de um fenômeno terrestre, tem-se que avaliar os valores dos correspondentes atributos para as diferentes épocas em foram feitos os levantamentos. Este processo é realizado em três passos. O primeiro é identificar as áreas de levantamento que contêm sobreposição. O segundo é identificar o fenômeno nas diferentes épocas e o último passo é determinar a variação temporal a partir dos correspondentes valores dos atributos. Assim, se não existir sobreposição entre áreas de levantamento então não existe variação temporal.

Para analisar os tipos de relacionamentos espaciais entre áreas de levantamento está-se supondo dois objetos "**A"** e "**B"** do tipo "AreaLevantamento" e para estes se pode identificar os seguintes relacionamentos:

a) **A** é disjunto de **B**;

b) **A** toca **B**;

c) **A** é adjacente de **B**;

d) **A** igual a **B**;

e) **A** intercepta **B**;

f) **A** coincide com **B**;

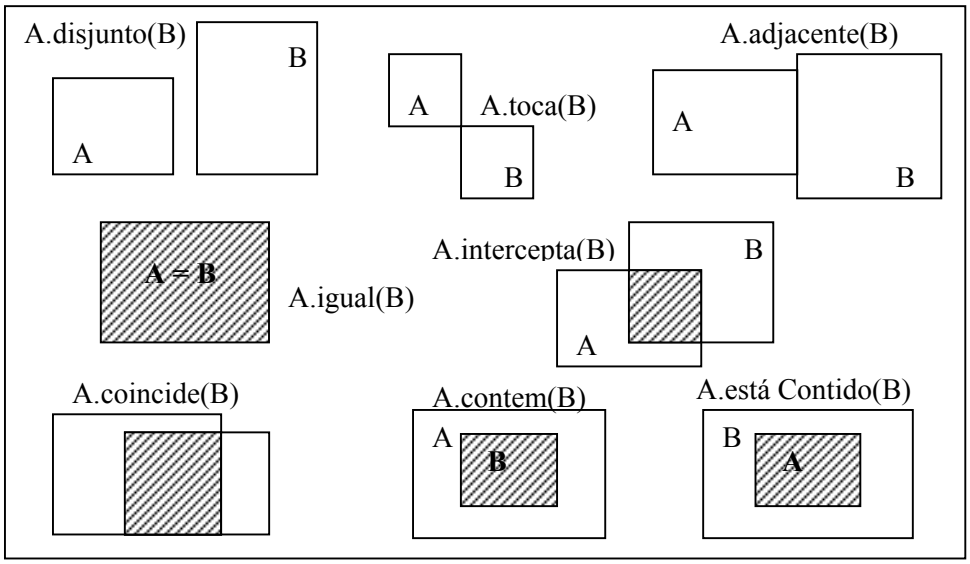
g) **A** contém **B;**

h) **A** está contido em **B**.

Na fig. 44 são apresentados esses tipos de relacionamentos espaciais que são recíprocos entre si. Por exemplo: se **A** é disjunto de **B**, então **B** é disjunto de **A**. Exceção apenas para os relacionamentos "contém" e "está contido". Os relacionamentos "disjunto", "toca" e "adjacente" implicam na inexistência de recobrimento e, conseqüentemente, não pode-se avaliar se um fenômeno apresenta ou não variação temporal. Todos os outros tipos de relacionamento implicam em

sobreposição das áreas de levantamento. Entretanto, isto não é suficiente para avaliar se um mesmo fenômeno apresenta ou não variação temporal, porque o mesmo fenômeno pode não ter sido levantado.

FIGURA 44 - TIPOS DE RELACIONAMENTOS ESPACIAIS ENTRE OBJETOS "AreaLevantamento"



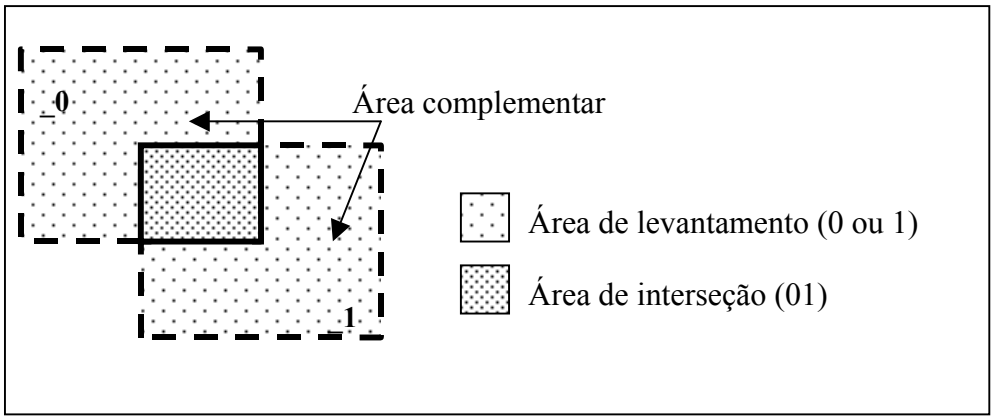
Como os objetos do tipo "AreaLevantamento" são instanciados somente a partir de um levantamento existente, concebeu-se os métodos da classe "AreaLevantamento" de modo que não implicassem em retornar um objeto "AreaLevantamento" como resultado de uma operação entre objetos deste tipo. Dessa forma foram implementados os métodos: "disjunto", "toca", "adjacente", "igual", "intercepta", "coincide", "contém" e "está contido" para a classe "AreaLevantamento" tendo como retorno um valor falso ou verdadeiro. Assim é feita a identificação do tipo de relacionamento entre objetos "AreaLevantamento" e, em seguida, se invoca uma operação que se realiza sobre os retângulos envolventes de cada área de levantamento.

3.5.2 Delimitação dos Relacionamentos Espaciais entre "AreaLevantamento"

Inicialmente, considerou-se que para lidar com objetos do tipo "AreaLevantamento" seria suficiente operar sobre os retângulos que envolvem cada área de levantamento, porque o que interessava era determinar a variação temporal que está restrita às áreas de levantamento com sobreposição. O inconveniente desta

hipótese é que o modelo somente representaria as áreas de interseção e, com isto, não seria possível, tratar-se os relacionamentos que são complementares à interseção, que são igualmente importantes. Por exemplo, sejam os levantamentos (0) e (1), pode-se desejar conhecer a área que está contida no levantamento-0, mas não no levantamento-1, ou vice-versa. Isto é análogo à operação diferença entre conjuntos. Como se pode constatar na fig. 45, essas áreas complementares são delimitadas por polígonos ao invés de retângulos. Como conseqüência, teve-se de conceber a classe "Poligono".

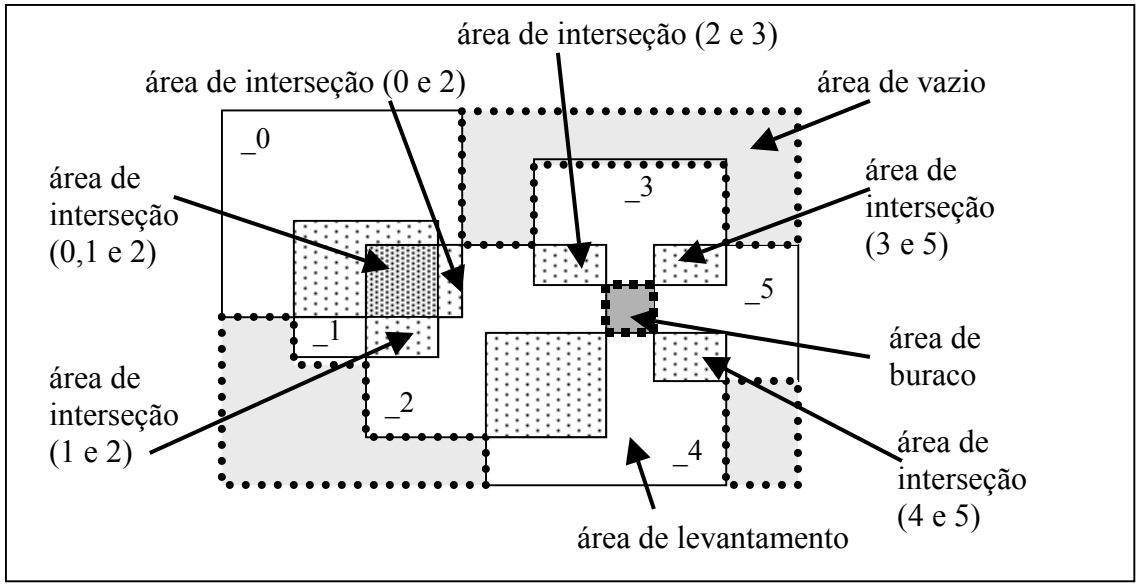
FIGURA 45 - RELACIONAMENTOS COMPLEMENTARES À INTERSEÇÃO



Na fig. 46 é apresentado um exemplo com seis levantamentos que são identificados pelos índices de zero até cinco. Como resultado final dos possíveis relacionamentos espaciais entre as áreas de levantamento pode-se identificar áreas que são caracterizadas por apresentar sobreposição de levantamentos e áreas que são caracterizadas apenas por um único levantamento sem ter, portanto sobreposição. A partir destas áreas, que são delimitadas por polígonos, é possível se obter uma que represente a união de todas as áreas de levantamento e o correspondente retângulo que a envolve.

Além disso, é possível identificar também áreas que não têm qualquer levantamento e, neste caso, existem dois tipos que estão sendo denominadas de áreas de vazio e áreas de buraco. Uma área de vazio é aquela que representa o complemento da área de união em relação ao retângulo envolvente. A área de buraco é aquela sem recobrimento e que está interna à área união. Este tipo de área pode surgir à medida que novos levantamentos são realizados (fig. 46).

FIGURA 46 - EXEMPLO SIMULANDO VÁRIOS LEVANTAMENTOS



Para organizar o conjunto de polígonos, fez-se um agrupamento desses segundo dois tipos: polígonos básicos e polígonos secundários. Foram agrupados como polígonos básicos o tipo união, o tipo interseção e o tipo buraco e como polígonos secundários todos os polígonos que são obtidos com uma operação análoga à diferença entre conjuntos, neste grupo estão todos os polígonos complementares e o polígono envolvente. O polígono do tipo vazio é um caso particular de polígono complementar, ou seja, é o complemento do polígono do tipo união em relação ao polígono envolvente.

Para representar as áreas delimitadas por polígonos, concebeu-se a classe "Polígono" que é constituída pelos seguintes membros: um conjunto de pontos do polígono e um retângulo que delimita o polígono. Para identificar o tipo de polígono se adotou um membro que pode assumir os seguintes valores: "Intersecao"; "Buraco"; "Uniao"; "Complementar"; "Envolvente"; e "Vazio", que são os tipos de relacionamentos inicialmente identificados quando dois objetos do tipo "Poligono" se relacionam. Para caracterizar o número de áreas de levantamento numa sobreposição, adotou-se um membro numérico que identifica quantos levantamentos existem para o polígono em questão. Entretanto, como o número de levantamentos está associado, de forma única, com o instante de observação, denominou-se este membro de temporalidade, sendo que este pode assumir os valores apresentados na tab. 2.

TABELA 2 - TIPOS DE TEMPORALIDADES ASSOCIADAS COM UM OBJETO “Polígono”

Temporalidade	Levantamentos
-1	nenhum
0	existe um único
1	existem dois
2	existem três
.....	.....
n-1	existem n

3.5.3 Recobrimento e Classe “Recobrimento”

Para representar todos os relacionamentos espaciais entre os objetos da classe “AreaLevantamento”, que são gerados à medida que novos levantamentos são realizados, criou-se o conceito de recobrimento. Em função disto, concebeu-se a classe “Recobrimento”. Para caracterizar um objeto desta classe foram previstos os seguintes membros: um conjunto de polígonos básicos; um conjunto de polígonos secundários; um retângulo que envolve toda a área de recobrimento; e um identificador do recobrimento; e um identificador das áreas de levantamento que deram origem ao recobrimento (fig. 47).

FIGURA 47 - MEMBROS DA CLASSE “Recobrimento”

Recobrimento
poligonosBasicos poligonosComplementares retanguloEnvolvente numeroRecobrimento areasLevantamento

As principais responsabilidades da classe “Recobrimento” dizem respeito com os relacionamentos que podem se verificar entre os objetos do tipo “AreaLevantamento”. Em função disto se pode obter informação sobre:

a) as áreas de levantamento;

b) as áreas sem levantamento;

c) as áreas com dois ou mais levantamentos;

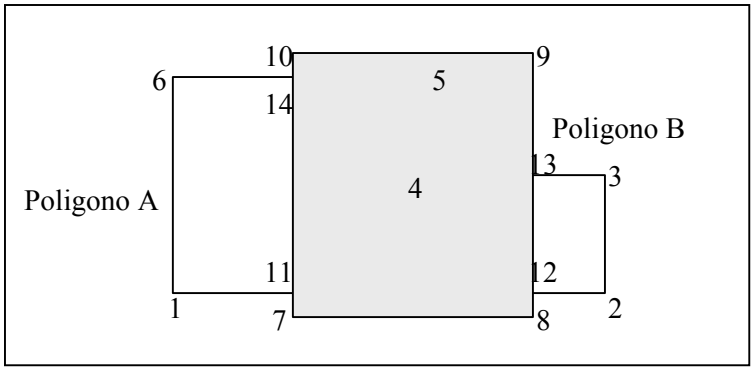
d) toda a área de recobrimento;

e) o tipo de relacionamento entre uma nova “AreaLevantamento” e um dos recobrimentos existentes.

3.5.4 Métodos para Obter os Polígonos Básicos e Complementares

Para obter os polígonos básicos (união, interseção e buraco) e secundários (complementares e envolvente) que resultam do relacionamento espacial entre objetos do tipo “AreaLevantamento” foi necessário conceber e implementar métodos próprios denominados “poligonosBasicos” e “poligonosSecundários”. Seja o Polígono A com os vértices: [1, 2, 3, 4, 5 e 6] e o Polígono B como os vértices: [7, 8, 9 e 10] que se interceptam e geram o Polígono C (fig. 48) com os vértices: [11, 12, 13, 4, 5 e 14].

FIGURA 48 - INTERSEÇÃO ENTRE OS POLÍGONOS “A” E “B”



3.5.4.1 Descrição do algoritmo para obter os polígonos básicos e complementares

Inicialmente, são calculados os vértices de interseção entre os polígonos A e B [11, 12, 13 e 14] e estes são inseridos de forma ordenada nas seqüências de vértices dos polígonos A e B. Junto com a identificação do vértice se inclui o sinal “-“ para destacar que estes são pontos de interseção. Assim, após os polígonos A e B serem analisados quanto à interseção, as seguintes seqüências de vértices são obtidas: polígono A [1,-11, -12, 2, 3, -13, 4, 5, -14 e 6] e polígono B [7, 8, -12, -13, 9, 10, -14 e –11].

Tomando-se por base cada ponto de interseção e caminhando-se de forma alternada entre as seqüências de vértices de A e B, no sentido da esquerda para a direita, são extraídas as seqüências de vértices que estão compreendidas entre os

pontos de interseção. Para o exemplo da fig. 48, então se teria:

a) o algoritmo inicia com o primeiro vértice de interseção sobre a seqüência de vértices do polígono A que é [-11];
b) como o próximo vértice na seqüência é um vértice de interseção [-12] então se obtém a seqüência [-11, -12] e alterna-se para o polígono B;
c) a partir da seqüência de B e iniciando-se no vértice [-12] complementa-se a seqüência com o vértice [-13], obtendo-se assim: [-11, -12, -13];
d) alternando-se para a seqüência de A e iniciando-se no vértice [-13] obtém-se a seqüência: [-11, -12, -13, 4, 5, -14];
e) alternando-se para a seqüência de B se chega ao final do processo porque o vértice (-11) é igual ao vértice inicial [-11]. Assim a seqüência resultante tendo por base o vértice [-11] é: [-11, -12, -13, 4, 5 e -14] . Esta seqüência representa o polígono **C**, como se pode constatar na fig. 48.

Esse mesmo procedimento deve ser repetido para cada um dos outros vértices de interseção:

a) tendo-se por base o próximo vértice de interseção [-12] e caminhando-se sobre a seqüência de vértices de A, no sentido da esquerda para a direita, obtém-se a seqüência: [-12, 2, 3 e -13];
b) Alternando-se para o polígono B, complementa-se a seqüência com os vértices: [-12, 2, 3, -13, 9, 10 e -14];
c) alternando-se para A**,** chega-se à seqüência: [-12, 2, 3, -13, 9, 10, -14, 6, 1 e -11];
d) alternando-se para B**,** chega-se à seqüência: [-12, 2, 3, -13, 9, 10, -14, 6, 1, -11, 7 e 8], que representa o polígono que é a união de A e B, como se pode constatar na fig. 48;

Repetindo-se os mesmos procedimentos para os vértices de interseção restantes [-13] e [-14], obtêm-se as seqüências: [-13, 4, 5, –14, -11 e -12] e [-14, 6, 1, -

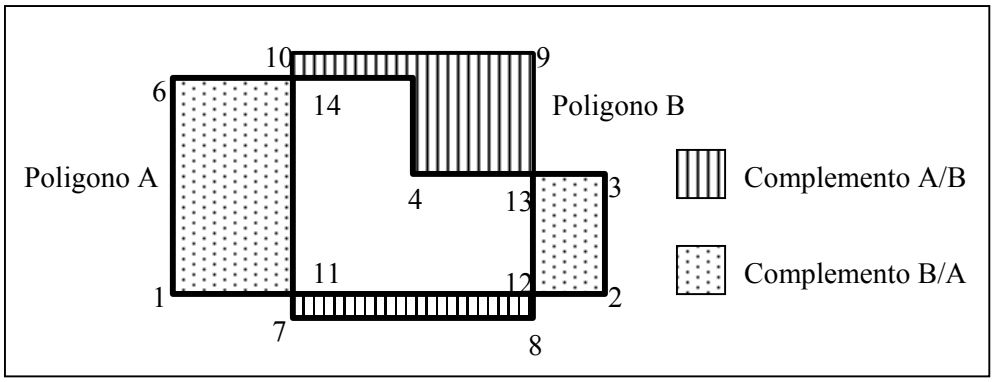
11, 7, 8, -12, 2, 3, -13, 9 e 10] que são iguais, respectivamente, aos polígonos interseção e união, já obtidos anteriormente, item (e) e item (d).

Com os procedimentos descritos anteriormente, somente é possível descobrir-se os polígonos do tipo interseção, união e buraco, que estão sendo considerados como polígonos básicos. Entretanto, quanto aos polígonos secundários, torna-se necessário um outro procedimento. Após uma série de experimentos, descobriu-se que quando se alterna de um polígono para outro e faz-se a pesquisa usando o sentido trocado, ou seja, se a pesquisa sobre A é feita da esquerda para a direita, então em B deve ser feita da direita para esquerda. Com isto, consegue-se obter os polígonos complementares à interseção. Então, com base nas seqüências de vértices: A [1,-11, -12, 2, 3, -13, 4, 5, -14 e 6] e B [7, 8, -12, -13, 9, 10, -14 e -11] e:

a) usando-se como vértice inicial [-11] e pesquisando-se da esquerda para a direita sobre a seqüência de A**,** obtém-se a seqüência: [-11, -12];
b) alternando-se para B e pesquisando a partir do vértice [-12] da direita para a esquerda obtém-se a seqüência: [-11, -12, 8, 7]. Como o vértice [-11] é igual ao vértice inicial o processo termina. Como se pode verificar na fig. 49 a seqüência obtida representa um dos polígonos complementares de A em relação à B;
c) repetindo-se o mesmo procedimento para o vértice de interseção [-12] sobre o polígono A obtém-se a seqüência: [-12, 2, 3, -13]. Esta seqüência representa um outro polígono que é o complemento de B em relação à A (fig. 49).
d) repetindo-se este procedimento para o vértice [-13] sobre o polígono A obtém-se a seqüência: [-13, 4, 5, -14];
e) alternando-se para o polígono B e pesquisando-se da direita para esquerda obtém-se a seqüência: [-13, 4, 5, -14, 10, 9], que representa o outro polígono que é o complemento de A em relação à B (fig. 49).
f) repetindo-se o processo para o vértice [-14] obtém-se: [14, 6, 1, -11];
g) alternando-se para B e pesquisando-se da direita, chega-se ao vértice

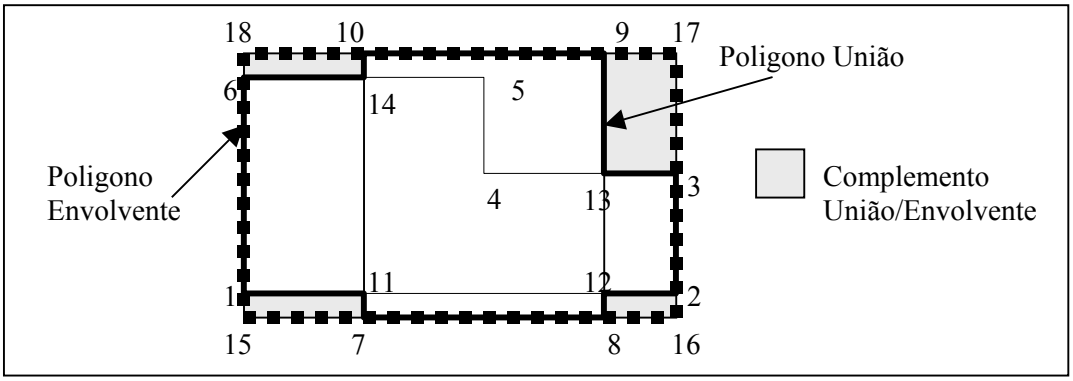
inicial e o processo termina. Como se pode constatar na fig. 49, a seqüência obtida representa o outro polígono complementar, neste caso o complemento de B em relação à A.

FIGURA 49 - COMPLEMENTO DE "A" EM RELAÇÃO À "B" E VICE-VERSA



Para complementar a família de polígonos resta somente o polígono envolvente e os polígonos do tipo vazio. O polígono envolvente é gerado diretamente com base nos valores de máximo e mínimo, ou seja, a partir dos valores de coordenadas x-mínimo e y-mínimo entre os vértices existentes, inicializa-se o canto inferior esquerdo, e com os valores de x-máximo e y-máximo, inicializa-se o canto superior direito. Além disto, é possível perceber-se que os polígonos vazios podem ser obtidos como polígonos complementares do polígono união em relação ao polígono envolvente (fig. 50).

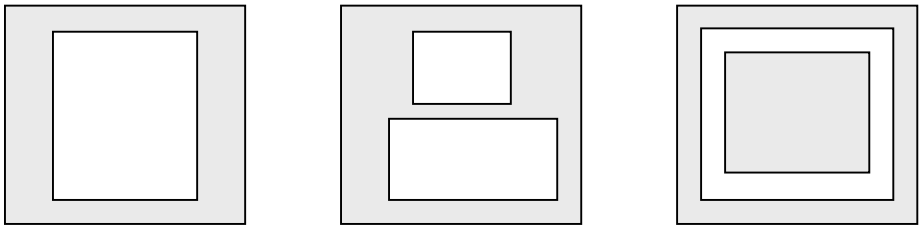
FIGURA 50 - RELAÇÃO ENTRE POLÍGONO UNIÃO E POLÍGONO ENVOLVENTE



Durante a realização de alguns experimentos, percebeu-se que para os relacionamentos "está contido" ou "contém", obtém-se um tipo de polígono distinto

dos já apresentados. Por exemplo, suponha que um polígono A contém um polígono B, então, tem-se como resultado deste relacionamento dois polígonos, ou seja, um que é do tipo interseção e é igual ao polígono B e um outro polígono que envolve o polígono interseção é igual ao polígono A menos o polígono B (fig. 51a). Inicialmente, para este polígono, deu-se o nome de "Polígono Vazado" uma vez que este apresenta um orifício. Posteriormente, percebeu-se que, em situações particulares, o polígono vazado pode ter mais do que um elemento interno (fig. 51b). Além disto, o polígono vazado pode tornar-se aninhado (fig. 51c).

FIGURA 51 - POLÍGONO VAZADO E POLÍGONO VAZADO E ANINHADO



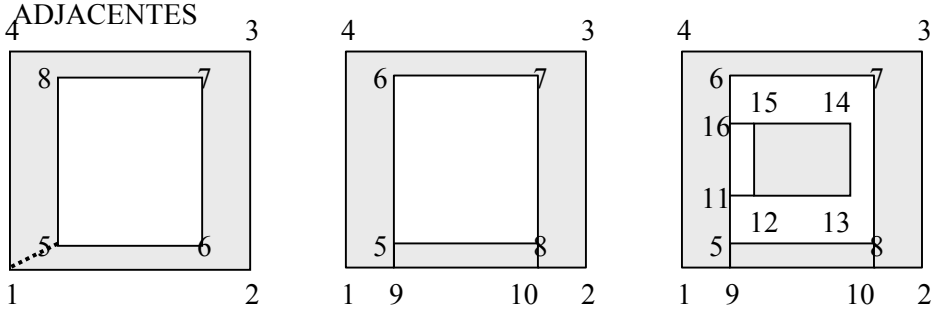
a) Polígono vazado

b) Polígono vazado com dois orifícios

c) Polígono vazado e aninhado

Em conseqüência disso, foi reavaliada a solução implementada para lidar com relacionamentos do tipo "contém" ou "está contido" e percebeu-se que qualquer polígono vazado pode ser decomposto em dois polígonos adjacentes. Por exemplo, seja o polígono vazado [1, 5, 8, 7, 6, 5, 1, 2, 3, 4] (fig. 52a) é possível constatar que este pode ser decomposto em dois polígonos simples e adjacentes com as seguintes seqüências de vértices [1, 9, 5, 6, 7, 8, 10, 2, 3, 4] e [9, 10, 8, 5] (fig. 52b). De forma análoga, pode-se aplicar o mesmo princípio para o polígono vazado e aninhado (fig. 52c). Com isto, eliminou-se o caso particular de polígono vazado.

FIGURA 52 - DECOMPOSIÇÃO DO POLÍGONO VAZADO EM DOIS POLÍGONOS ADJACENTES



a) Polígono vazado

b) Polígono vazado decomposto

c) Decomposição do polígono vazado e aninhado

3.5.4.2 Seleção da área com recobrimento

Com base nos objetos do tipo “Recobrimento”, idealizou-se que uma certa área pudesse ser selecionada e para esta seria então determinada a variação temporal. Para isto, foram desenvolvidos três métodos. No primeiro, gerou-se uma tabela em que são registrados todos os tipos de relacionamentos entre os objetos “AreaLevantamento”. Estes relacionamentos podem ser: “adjacente”, “coincide”, “contém”, “está contida”, “disjunta”, “igual”, “toca” e “intercepta” (ver item 3.5.1). No exemplo apresentado a seguir, foi simulado que existem oito áreas de levantamento e que os tipos de relacionamentos entre estas áreas de levantamento são identificados pelas seguintes abreviaturas: adjacente =”adjc”; coincide =”coin”; contém =”cntm”; está contida =”ctda”; disjunta +”disj”; igual =”igul”; toca =”toca”; e interseção = “intr”. Na tab. 3 é apresentado o conteúdo que foi gerado a partir da identificação do tipo de relacionamento entre os objetos “AreaLevantamento”.

Assim que uma “AreaLevantamento” e um tipo de relacionamento são escolhidos todas as “AreaLevantamento” que satisfazem este tipo de relacionamento são selecionadas e ordenadas para calcular-se a variação temporal. A principal restrição quanto a este método está na imposição de um único tipo de relacionamento entre as áreas de levantamento. Em função disto, foi realizado um segundo experimento em que fosse possível escolher o percentual de sobreposição a partir do qual conjuntos de objetos “AreaLevantamento” são selecionados e ordenados para determinar-se a variação temporal.

TABELA 3 - RELACIONAMENTOS ENTRE OS OBJETOS “AreaLevantamento”

	“AreaLevantamento”							
	área_0	área_1	área_2	área_3	área_4	área_5	área_6	área_7
área_0	xxxx	intr	intr	disj	igul	coin	intr	disj
área_1	intr	xxxx	intr	disj	intr	disj	coin	Intr
área_2	intr	intr	xxxx	adjc	intr	coin	disj	Intr
área_3	disj	disj	adjc	xxxx	disj	adjc	disj	intr
área_4	igul	intr	intr	disj	xxxx	coin	intr	disj
área_5	coin	disj	coin	adjc	coin	xxxx	intr	disj
área_6	intr	coin	disj	disj	intr	intr	xxxx	intr
área_7	disj	intr	intr	intr	disj	disj	intr	xxxx

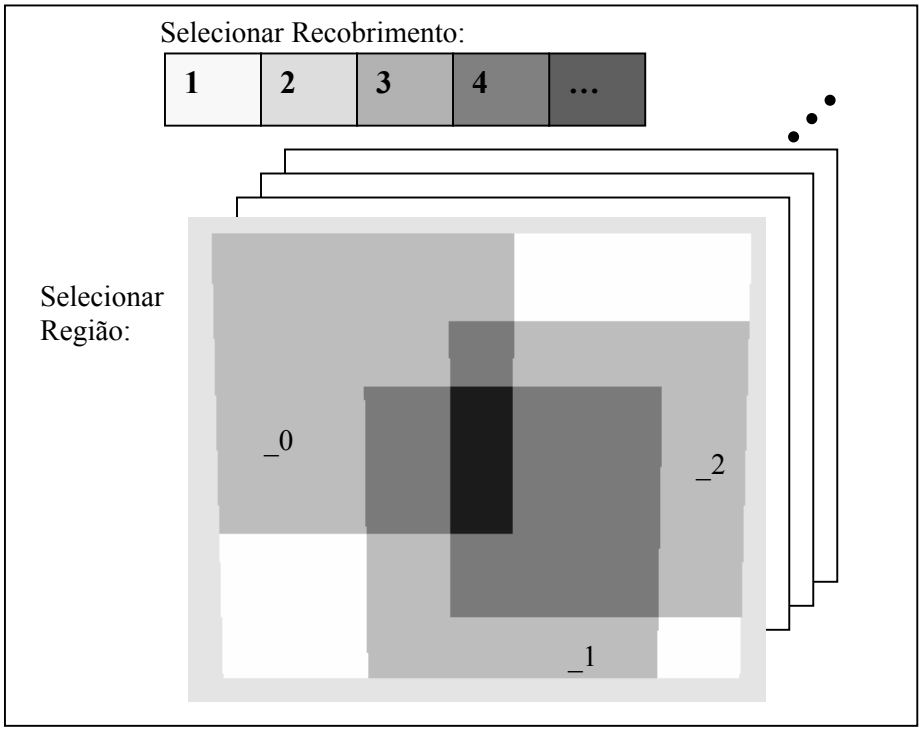
Nesse segundo método é possível se selecionar uma percentagem de sobreposição, dentre um conjunto de alternativas pré-definidas. No caso em que a sobreposição entre áreas de levantamento é inferior à percentagem selecionada, então a área de levantamento não é considerada para determinar a variação temporal. O que se pretende com isto, é primeiramente filtrar as regiões que têm uma sobreposição abaixo de um certo limiar e, ao mesmo tempo, possibilitar que sejam selecionados grupos de áreas de levantamento. Com esta abordagem, ganhou-se generalidade em relação ao primeiro método. Por exemplo: se for selecionado um percentual de sobreposição de 99%, então serão pesquisados somente os objetos "AreaLevantamento" que têm áreas de sobreposição maior ou igual a 99%, o que corresponderia num contexto prático a um relacionamento de igualdade. Na fig. 53 é apresentada a janela utilizada para seleção do percentual de sobreposição.

FIGURA 53 - JANELA PARA O USUÁRIO SELECIONAR O LIMIAR DE SOBREPOSIÇÃO ENTRE ÁREAS DE LEVANTAMENTO

No terceiro método a idéia básica foi disponibilizar o conjunto de polígonos obtidos para um certo recobrimento e permitir que, interativamente, seja selecionada uma região de interesse. Para isto, foi implementado um aplicativo em Java que apresenta esta família de polígonos para as diferentes épocas (fig. 54). Uma vez selecionada uma região de interesse, o sistema determina a variação temporal para a

superfície terrestre a partir dos levantamentos e disponibiliza isto para ser visualizado na forma de uma animação segundo o padrão VRML.

FIGURA 54 - VISUALIZAÇÃO DOS RELACIONAMENTOS ENTRE OS DISTINTOS LEVANTAMENTOS REALIZADOS COM O PASSAR DO TEMPO



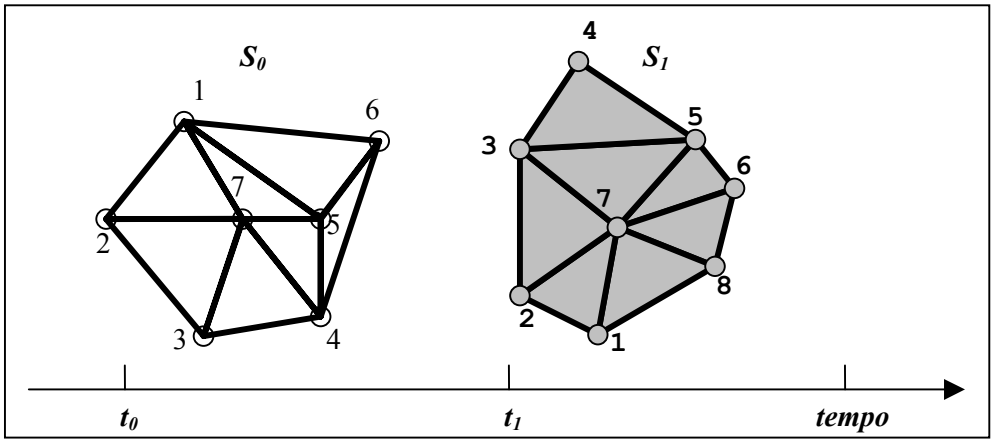
3.5.4.3 Determinação da variação temporal

A partir de uma área selecionada é possível então determinar a variação temporal sofrida por um fenômeno terrestre, avaliando-se os valores dos correspondentes atributos nas diferentes épocas. Para a realização dos experimentos sobre a determinação da variação temporal, foi escolhido como fenômeno terrestre a superfície topográfica uma vez que os objetos deste tipo apresentam o maior grau de complexidade quanto à caracterização da superfície, por ser esta irregular e tridimensional. Além disto, para determinar a variação temporal para objetos deste tipo estão envolvidos os processos de busca e de interpolação para cada vértice da superfície topográfica nas diferentes épocas.

A partir de um levantamento topográfico é possível reconstituir, representar e, se desejado, visualizar uma porção da superfície topográfica *S*. Um modelo de representação bastante usado e conhecido é o diagrama de Delaunay. Com este é

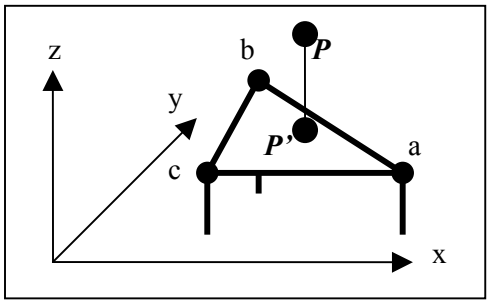
possível representar a estrutura espacial da superfície *S* por meio de um conjunto de faces triangulares planas, que satisfazem ao critério do círculo vazio (OKABE, BOOTS, SUGIHARA, 1995, p. 94). Quando novos levantamentos sobre a mesma região são realizados, tem-se então várias superfícies S_t, em que *t* (0,1,2,3,...) representa o instante em que cada levantamento foi realizado (fig. 55).

FIGURA 55 - LEVANTAMENTOS REALIZADOS COM O PASSAR DO TEMPO



Para se determinar a variação temporal da superfície topográfica é necessário que se conheça a altitude de cada vértice *P* para cada instante *t* sobre as distintas superfícies S_t. Para isto, tem-se que resolver um sistema formado pela equação do plano que passa por três pontos, que são os vértices da face triangular, e a equação da reta que passa por *P* e é ortogonal ao plano xy (fig. 56).

FIGURA 56 - A INCÓGNITA É O VALOR DA ALTITUDE DE *P'*, INTERSEÇÃO DO PLANO (abc) COM A RETA ORTOGONAL AO PLANO (XY) QUE PASSA POR *P*



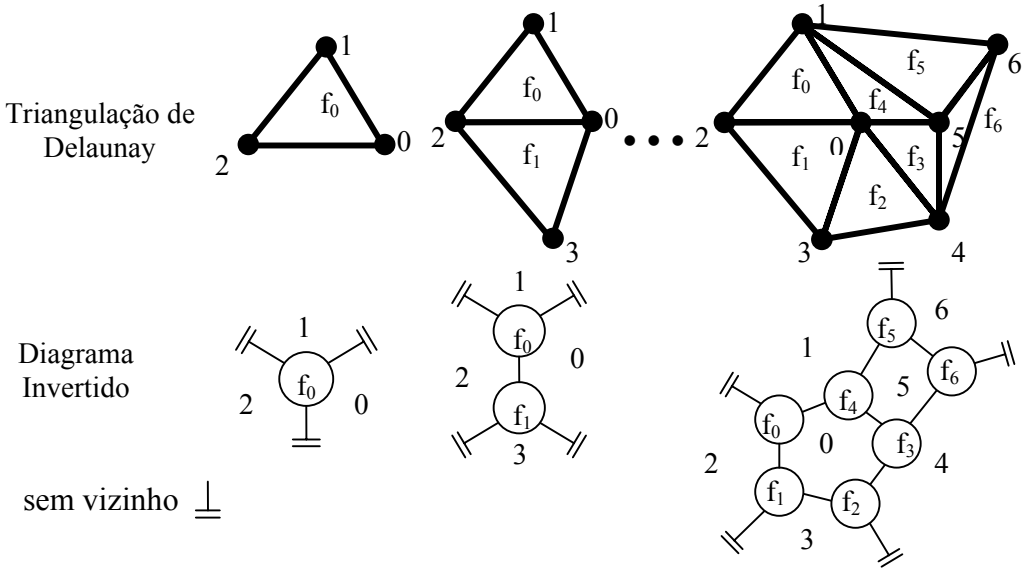
Em princípio não existe dificuldade para se resolver esse sistema de equações lineares. Entretanto, tem-se que identificar qual é a face triangular da superfície S_t que contém o ponto *P'*, projeção ortogonal de *P* sobre o plano xy. Esta é

uma questão típica de busca e pode ser crítica, porque o tempo despendido neste processo pode, no pior dos casos, crescer à medida que o volume de dados aumenta, o que é normalmente o caso de representações temporais da superfície topográfica. Para resolver este problema foi idealizado um mecanismo para acelerar a busca.

3.5.4.4 Mecanismo concebido para acelerar o processo de busca

O mecanismo concebido, para acelerar o processo de busca, é composto por dois elementos. O primeiro é uma lista em que estão armazenados os índices j dos vértices v_j da superfície S_t, que representam as distâncias, ordenadas de forma crescente, para o vértice P_i da superfície S_u. Assim, os primeiros elementos da lista identificam os vértices v_j mais próximos de P_i. O segundo elemento é denominado de Diagrama Invertido. Na realidade, este diagrama representa o conjunto de faces que são comuns a cada vértice v_j. A denominação de Diagrama Invertido foi dada porque na sua representação gráfica existe uma inversão em relação ao Diagrama de Delaunay, ou seja, o que são vértices e faces no Diagrama de Delaunay tornam-se, respectivamente, faces e vértices no Diagrama Invertido (fig. 57).

FIGURA 57 - ESTRUTURA DE DELAUNAY COM O DIAGRAMA INVERTIDO



3.5.4.5 Geração do Diagrama Invertido

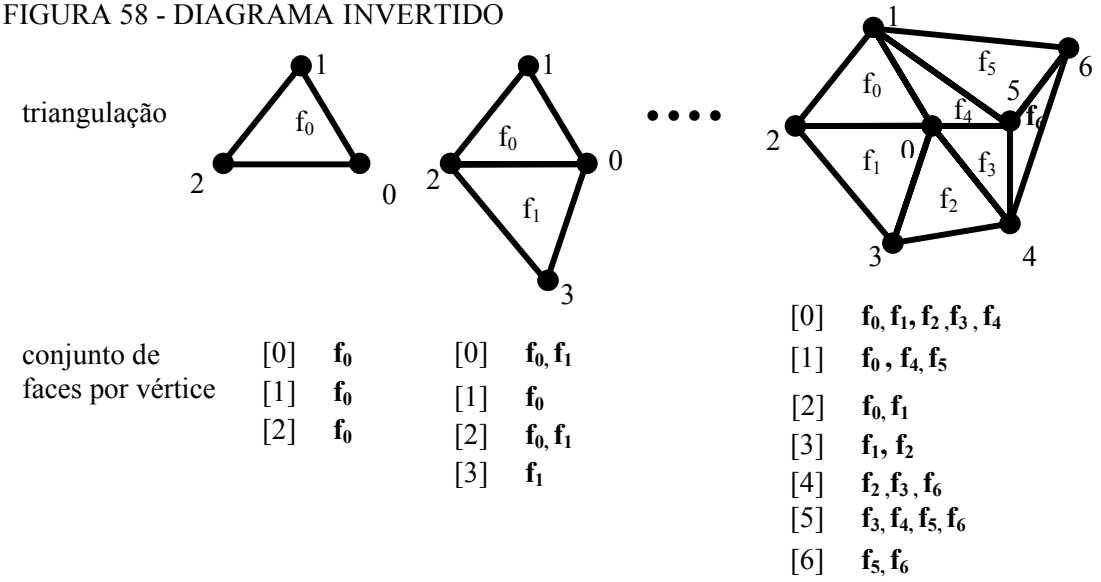
Para se gerar o Diagrama Invertido, em primeiro lugar, tem-se que instanciar a estrutura de Delaunay, que é uma classe que foi concebido e é composta pelos

seguintes membros:

a) conjunto de todos os vértices da estrutura Delaunay;
b) conjunto de todas as faces da estrutura Delaunay;
c) conjunto dos vértices da envolvente em ordem seqüencial;
d) conjunto das faces da envolvente;
e) coordenadas (x,y,z) do retângulo que envolve a estrutura de Delaunay.

Os dados necessários para instanciar a estrutura de Delaunay são obtidos quando se realiza a triangulação de Delaunay. Para isto, está-se utilizando o programa "***Triangle***" (ver item 4.3 PROGRAMA *TRIANGLE*). Como resultado da triangulação de Delaunay são obtidos dois conjuntos de dados contendo, respectivamente, todos os vértices e todas as faces da estrutura de Delaunay. A partir destes conjuntos, pode-se inicializar diretamente os membros descritos, anteriormente, nos itens (a) e (b), que são os conjuntos de vértices e faces da estrutura de Delaunay. Para os outros dois conjuntos descritos nos itens (c) e (d), respectivamente, os vértices e as faces da envolvente, são necessários pré-processamentos porque embora estejam explicitados quais são os vértices que pertencem à envolvente (código numérico igual um), estes não estão ordenados seqüencialmente. Além disto, todos os vizinhos das faces têm de ser inicializados. O último membro, descrito no item (f), é inicializado com base nas coordenadas: (x-mínimo; y-mínimo) e (x-máximo; y-máximo) dentre os vértices da estrutura de Delaunay. Após terem sido realizados esses pré-processamentos então é instanciada a estrutura de Delaunay. A partir deste ponto, pode-se gerar o Diagrama Invertido, analisando-se cada face da estrutura de Delaunay e as suas vizinhanças (fig. 58).

FIGURA 58 - DIAGRAMA INVERTIDO



conjunto de faces por vértice

[0]	**f_0**	[0]	**f_0, f_1**	[0]	**f_0, f_1, f_2, f_3, f_4**
[1]	**f_0**	[1]	**f_0**	[1]	**f_0, f_4, f_5**
[2]	**f_0**	[2]	**f_0, f_1**	[2]	**f_0, f_1**
		[3]	**f_1**	[3]	**f_1, f_2**
				[4]	**f_2, f_3, f_6**
				[5]	**f_3, f_4, f_5, f_6**
				[6]	**f_5, f_6**

O algoritmo tem início a partir do momento em que se seleciona o primeiro ponto P_i da superfície S_t e a superfície S_u. Inicialmente, é feito um teste para se verificar se P_i' é externo, ou está sobre alguma das arestas que formam o polígono envolvente a superfície S_u. Se P_i' for externo, então não será feita qualquer interpolação e adota-se para P_i' a altitude de P_i. Se P_i' estiver sobre alguma das arestas do polígono envolvente à superfície S_u, então é interpolado o valor da altitude de P_i' resolvendo-se o sistema formado pela equação da reta que passa pelos vértices que caracterizam a aresta e a equação da reta que passa por P e é ortogonal ao plano xy. Caso contrário, se P_i' é interno ao polígono envolvente de S_u, então se tem que gerar a lista de distâncias de P_i' para cada vértice v_j de S_u e usar o Diagrama Invertido.

A seguir, são apresentadas as etapas do algoritmo quando o ponto P_i' é interno ao polígono envolvente S_u:

a) selecionar a partir da lista de índices de distâncias o primeiro elemento, ou seja, o índice j que identifica o vértice v_j mais próximo de $P'_{i;}$

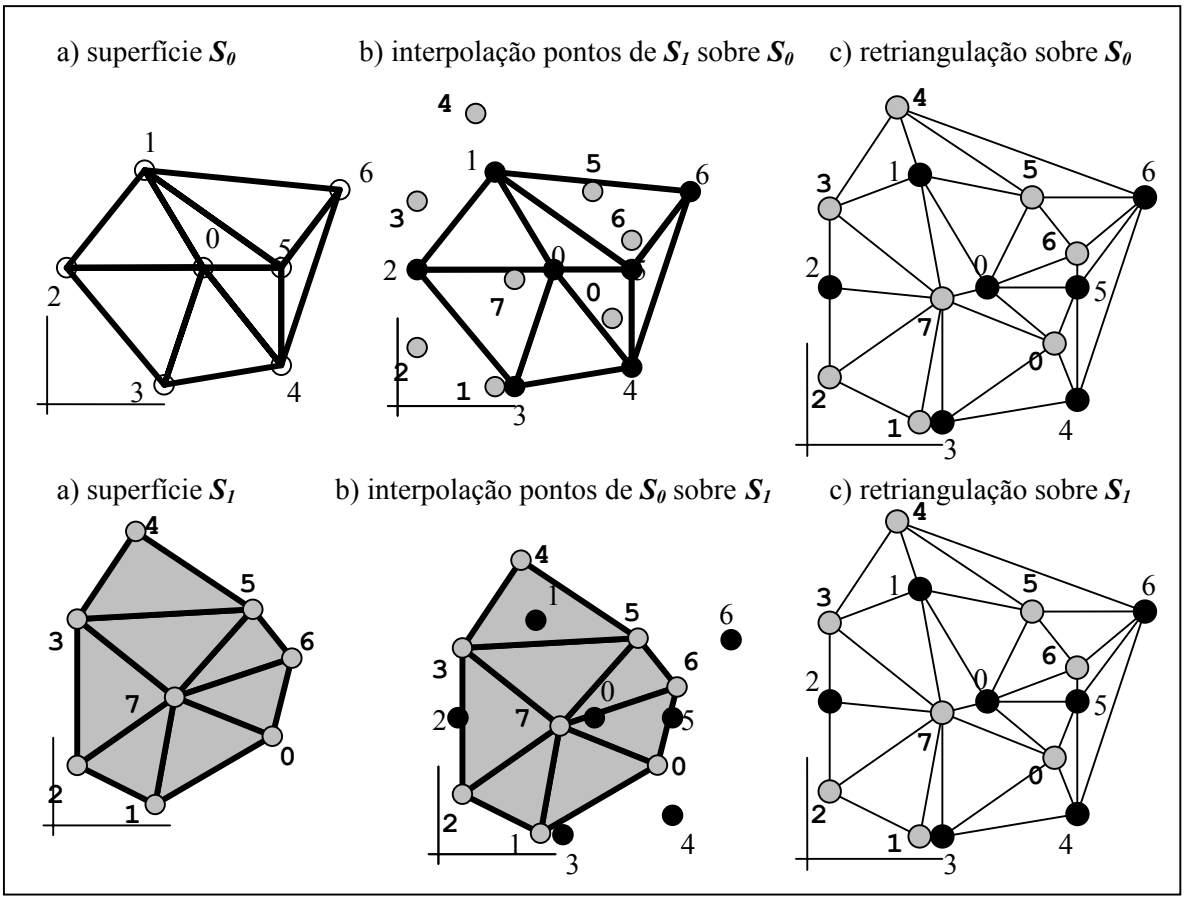
b) em seguida, busca-se no Diagrama Invertido o conjunto de faces comuns ao vértice v_j ;

c) para cada uma dessas faces é feito um teste para verificar se alguma dessas contém P_i';

d) caso nenhuma das faces testadas contenha P_i', então se seleciona o próximo índice j na lista de distâncias e volta-se a etapa (b) até que seja identificada a face que contém P_i'.

Terminada a identificação da face que contém P_i', então é interpolado o valor da altitude de P_i' resolvendo-se o sistema composto pela equação do plano, que contém os vértices da face identificada, e a equação da reta que passa por P e é ortogonal ao plano xy. No caso particular do ponto P_i' cair sobre uma das arestas do triângulo, então se calcula a altitude de P_i' determinando-se a interseção entre a equação da reta que passa pelos correspondentes vértices da aresta do triângulo e pela reta que passa por P_i e é ortogonal ao plano xy. Após ter sido realizado este processo de interpolação para todos os pontos sobre cada superfície, se refaz a triangulação de Delaunay para recompor as novas faces em cada superfície $\boldsymbol{S_t}$ (fig. 59).

FIGURA 59 - PROCESSO DE INTERPOLAÇÃO DOS PONTOS DA SUPERFÍCIE $\boldsymbol{S_0}$ SOBRE $\boldsymbol{S_1}$ E VICE VERSA



4 RECURSOS UTILIZADOS

Para o desenvolvimento desta tese foram utilizados os seguintes recursos: o ambiente de programação *NetBeans IDE*, o ambiente de modelagem *Together*, o programa *Triangle*, o programa MaxiCAD, o sistema Cortona e um micro computador. Os dados para a realização dos experimentos foram gerados por simulação a partir de uma carta digital 3D do centro politécnico da UFPR, na escala 1/2000.

4.1 AMBIENTE *NetBeans IDE*

O ambiente de programação utilizado é o *NetBeans IDE* 3.5.1, *Community Edition*, da *Sun Microsystems,* que pode ser obtido, livre de taxa, no endereço: http://java.sun.com/j2se/1.4.2/download.html. O *NetBeans IDE* tem um conjunto de recursos que permitem conceber, depurar, compilar e executar programas em *Java* (SUN, 2004). Tais programas podem ser do tipo aplicativo *Java* ou *applet Java*. Enquanto um aplicativo *Java* é executado utilizando um interpretador *Java,* um *applet Java* é um programa que é executado em uma página da Web por meio de um navegador Internet, como o *Netscape* ou o *Internet Explorer* (DEITEL; DEITEL, 2001, p. 119; CHAN; GRIFFTH; IASI, 1999, p. 1).

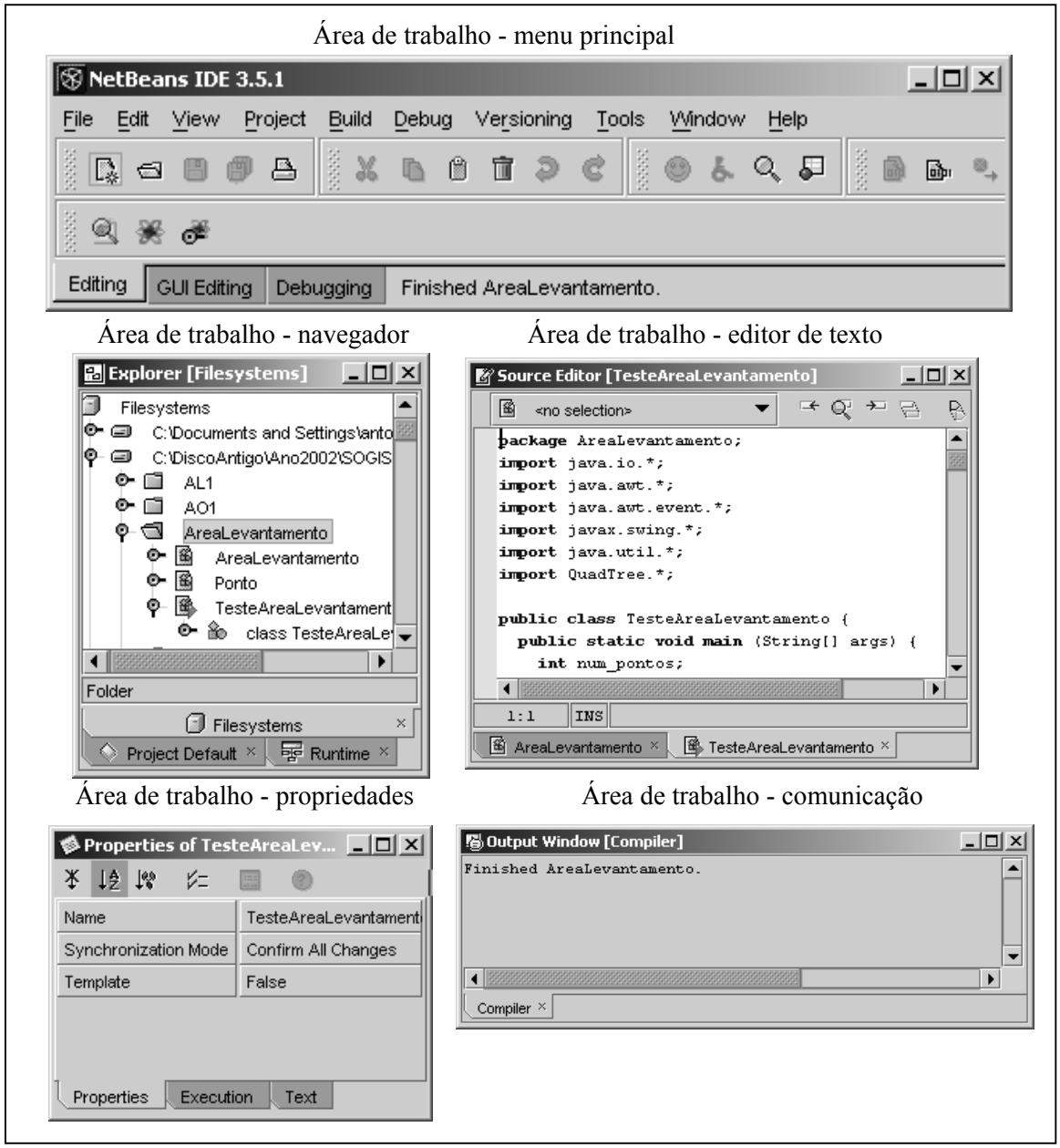
As áreas de trabalho que compõem o *NetBeans* recebem a denominação genérica de *IDE*, que é o acrônimo para *Integrated Development Environment*. O *IDE* é composto por cinco áreas de trabalho (fig. 60), que são:

a) a barra de menu principal - contém os recursos básicos para criar, abrir, salvar, compilar, depurar e executar um programa;
b) o navegador - permite navegar e selecionar os arquivos subdiretórios de sistema que estão montados segundo uma estrutura hierárquica. Para informar o estado do arquivo (sem compilação, com condição de erro e pronto para ser executado) são utilizados ícones ao lado da cada um;
c) o editor de arquivos fontes - é um editor de textos utilizado para criar o código fonte dos módulos implementados;

d) a área de propriedades - dá acesso aos valores das propriedades de cada nó que está selecionado;

e) a área de comunicação com o usuário - informa sobre o estado do sistema em função de alguma ação solicitada.

FIGURA 60 - AS CINCO ÁREAS DE TRABALHO QUE COMPÕEM O AMBIENTE



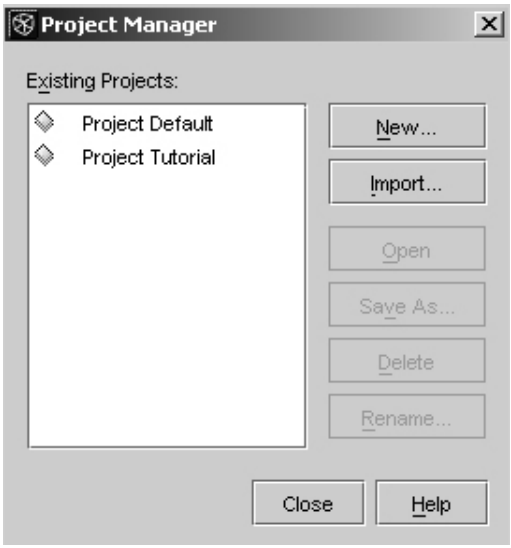
O ambiente *NetBeans IDE* foi importante durante as fases iniciais de análise e prototipação das classes e suas funcionalidades uma vez que só posteriormente se conseguiu o ambiente de modelagem *Together*. Dentre as várias tarefas que podem ser

realizadas com o ambiente *NetBeans IDE*, destacam-se: criar um projeto; montar um *filesystem*; criar um pacote; criar uma classe; criar um programa; compilar; e executar um programa. A seguir são descritos os principais passos necessários para cada uma destas tarefas.

4.1.1 Criar um Projeto

Para criar um novo projeto (ou então selecionar um projeto disponível) é utilizada a seqüência de comandos: *Project* > *Project Manager*, que são selecionados no menu principal. Com isto o sistema apresenta uma janela (fig. 61) em se pode introduzir o nome do projeto (ou então selecionar algum projeto existente).

FIGURA 61 - JANELA DE SELEÇÃO OU CRIAÇÃO DE PROJETO



4.1.2 Montar um *Filesystem*

O termo *Filesystem* caracteriza a hierarquia de diretórios e arquivos que estão disponíveis para serem utilizados. A partir do momento que um projeto tenha sido criado (ou selecionado) é apresentada à área de trabalho relativa ao navegador. Caso o projeto já tenha sido criado e os *filesystems* montados, então o sistema apresenta o conjunto de diretórios, subdiretórios e arquivos que estão disponíveis (*filesystems*). Se o projeto é novo, então somente após ter sido montado um (ou mais) *filesystem* é que é possível visualizar e acessar os subdiretórios e arquivos abaixo deste nó. Para montar um diretório de trabalho é utilizada a função *mount*, que está

disponível no ambiente *NetBeans IDE* a partir do menu principal na opção *File*. Procedimento para montar um *filesystem*:

a) selecionar sobre a barra de menu a opção *File*;
b) em seguida, *Mount Filesystem* > *Local Directory*;
c) clicar sobre a opção *Next*;
d) selecionar um diretório DiretorioTrabalho;
e) concluir clicando sobre a opção *Finish*;

Na fig. 62 é apresentada a área de trabalho do navegador, em que foi montado o *filesystem* c:\DiretorioTrabalho e na parte inferior, pode-se observar que está ativo o projeto Tese (*Project Tese*).

FIGURA 62 - EXEMPLO DE *filesystem* MONTADO

Como foram utilizadas também as classes do pacote GeoVRML, tem-se que montar este *filesystem*. Para isto se utiliza a seguinte seqüência de comandos:

a) selecionar sobre a barra de menu a opção *File*;
b) em seguida, *Mount Filesystem* > *Arquive Files*;
c) clicar sobre a opção *Next*;
d) selecionar um diretório em que esteja o "geovrml.jar";
e) concluir clicando sobre a opção *Finish*.

Na fig. 63 são apresentados os *filesystems* que foram montados, c:\DiretorioTrabalho e geovrml.jar[1].

[1] Um arquivo com extensão "jar" caracteriza um pacote de classes *Java* que estão compactadas.

FIGURA 63 - EXEMPLO DE *filesystem* E O PACOTE GeoVRML MONTADOS

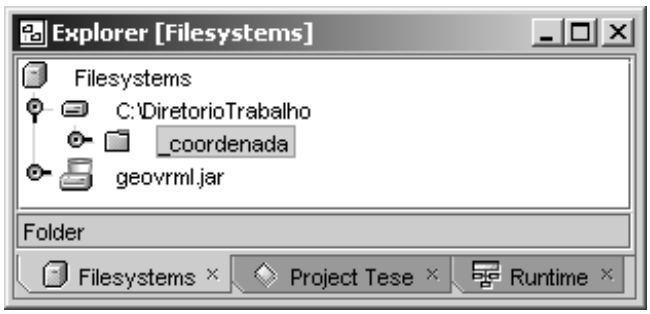
4.1.3 Criar um Pacote

O termo pacote, no contexto da linguagem de programação Java, é definido como "... uma coleção nomeada de classes (e possivelmente subclasses). Os pacotes servem para agrupar classes relacionadas e definem um *namespace*[2] para as classes que eles contêm" (FLANAGAN, 2000, p. 84). Para especificar um pacote é utilizada a palavra chave *package* no início de um documento Java. Após ter sido montado o diretório de trabalho, pode-se criar um pacote usando a seguinte seqüência de comandos:

a) selecionar o diretório em que se quer criar o pacote;
b) *File* > *New* > *Java Package* > *Next*;
c) Inserir o nome do pacote no campo *Name*: "_coordenada";
d) selecionar a opção: *Finish*;

Na fig. 64 é apresentado o *filesystem* ("c:\DiretorioTrabalho") que foi montado e o pacote ("_coordenada") que foi criado. Adotou-se o prefixo "_" para o nome de pacotes visando melhor diferenciar uma classe de um pacote de classes.

FIGURA 64 - EXEMPLO DO PACOTE "_coordenada"



[2] É uma coleção de nomes que são identificados por um *Uniform Resource Identifier* (URI - é uma seqüência compacta de caracteres usada para identificar um recurso na Web).

4.1.4 Criar uma Classe *Java* e Introduzir a Funcionalidade da Classe

A partir do momento que tenham sido montados os *fileSystems* e criados os pacotes é possível então criar cada classe que comporá o pacote. Procedimento para criar uma classe *Java*:

a) *File > New > Java Classes > Java Class > Next*

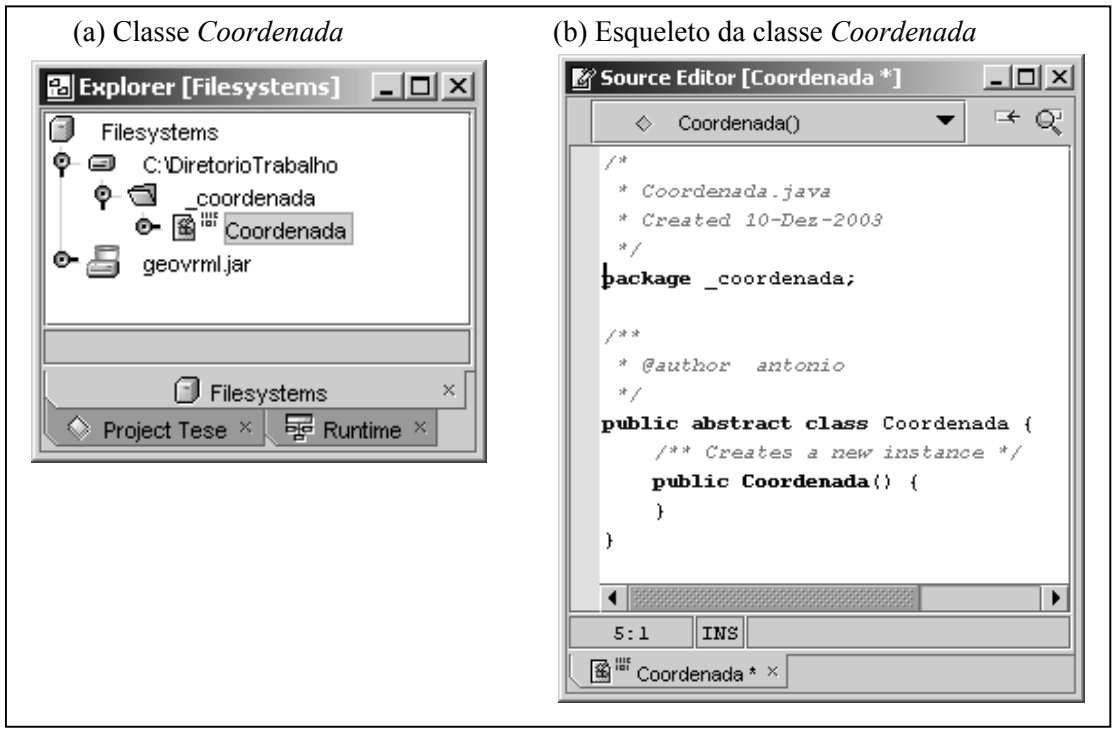
b) Inserir o nome da classe no campo *Name*: "Coordenada"

c) Selecionar o tipo de classe: *abstract*

d) Selecionar a opção: *Finish*

Agora o sistema apresenta o *filesystem* "DiretorioTrabalho" com o pacote "_coordenada" e a classe "Coordenada", que foi declarada como "*abstract*" (fig. 65a).

FIGURA 65 - EXEMPLO DA CLASSE COORDENADA



Na fig. 65b é apresentado o conteúdo do arquivo "Coordenada.java", que corresponde a classe abstrata "Coordenada". Pode-se observar a estrutura do construtor[3] que é automaticamente criada para a classe. O passo seguinte é introduzir a

[3] Construtor é um tipo de método usado para inicializar os objetos recém-criados (Flanagan, 2000, p. 66).

funcionalidade desejada para a classe. Como a classe Coordenada faz referência a objetos das classes "*coords*" e "*ellipsoid*", que fazem parte das classes concebidas pelo Grupo GeoVRML, é necessário primeiramente montar o arquivo "geovrml.jar", que contém tais classes, e declarar logo após a diretiva "*package*" que estas classes têm de ser importadas, usando para tanto a diretiva "*import*". Além disto, foram inseridas duas variáveis membro que identificam o tipo de elipsóide e o *datum* horizontal na classe Coordenada.

Na fig. 66 é apresentada parte do conteúdo da classe "Coordenada", em que estão declarados dois construtores, sendo um vazio, que inicializa o elipsóide e o datum horizontal relativos ao SAD-69. O outro construtor para ser chamado exige que se passe o tipo de elipsóide (um dos vinte e um tipos especificados pelo Grupo GeoVRML) e o datum horizontal. São também declarados métodos abstratos para conversão de coordenadas.

FIGURA 66 - FRAGMENTO DO CÓDIGO FONTE DA CLASSE COORDENADA

```
package _coordenada;
import geotransform.coords.*;
import geotransform.ellipsoids.*;
/**
  * @author  antonio
 */
public abstract class Coordenada {
    Ellipsoid e;
    String datumHrz;

 public Coordenada() {
        e =new SA_Ellipsoid();
        //inicializando (a,f) com elipsóide SAD-69
        datumHrz ="Chua";
    }
    public Coordenada(byte tipo,String datumH) {
        setElipsoide(tipo);
        datumHrz =datumH;
 }
 // metodos
```

FIGURA 60 - PUBLIC ABSTRACT VOID TOGCC(GCC_COORD_3D CGCC);

```
 public abstract void toGDC(Gdc_Coord_3d cGDC);
 public abstract void toUTM(Utm_Coord_3d cUTM);
 public void setElipsoide(byte tipo) {
   if(tipo == 0) e =new AA_Ellipsoid();
   else if(tipo == 1) e =new AM_Ellipsoid();
   else if(tipo == 2) e =new AN_Ellipsoid();
```

4.1.5 Compilar e Executar um Programa *Java*

Após ter sido inserida a funcionalidade desejada para a classe é possível então compilar e fazer os vínculos internos usando respectivamente as opções "*Compile*" e "*Build*" disponíveis na opção "*Build*" do menu. Para executar o aplicativo tem-se que selecionar "*Execute*" disponível em *Build* na barra de menu.

4.2 AMBIENTE *Together*

O ambiente de desenvolvimento utilizado para fazer a modelagem é chamado *Together* e foi desenvolvido pela *Borland*. Este ambiente proporciona um conjunto de recursos computacionais que permitem construir o sistema com base em modelos expressos segundo a UML (TOGETHER, 2004). Para se começar a modelar com o *Together* é necessário se criar um projeto. Isto consiste em definir um diretório raiz e um arquivo de projeto. No diretório raiz ficará armazenado o arquivo de projeto, todos os diagramas criados durante a modelagem e os arquivos de configuração. O arquivo de projeto tem o mesmo nome do projeto e uma extensão "tpr".

O ambiente *Together* é constituído por uma janela principal que é dividida em quatro regiões (fig. 67). Uma região para navegar, a qual permite que se tenha uma visão geral de toda a estrutura de projeto além de dar acesso a qualquer um dos módulos. Uma outra região para modelar e visualizar, em que são apresentados os elementos do modelo. Uma outra região que permite acessar e editar o código fonte dos elementos modelados. A última região, que fica na parte de baixo da janela principal informa ao usuário sobre a condição de uma tarefa específica que foi realizada.

Quando um novo projeto é criado o *Together* gera como diagrama *default*[4] um diagrama de classe e apresenta o conteúdo de projeto contido no diretório raiz. Na fig. 67 é dado um exemplo simples do ambiente *Together*. Na parte superior direita é visualizado um diagrama de classe com duas classes hipotéticas ("Class1" e "Class2"). Do lado esquerdo a estrutura de diretórios. Na parte de baixo, pode-se visualizar a estrutura do esqueleto de código, que é automaticamente gerada à medida que se vai

[4] Diagrama *default* é um diagrama que é gerado automaticamente pelo sistema caso o usuário não faça qualquer intervenção para especificar algo diferente.

modelando. Na fig. 68 é apresentado um exemplo de um diagrama *use-case* com dois atores que interagem através de módulo testar.

FIGURA 67 - AMBIENTE DE DESENVOLVIMENTO *Together*

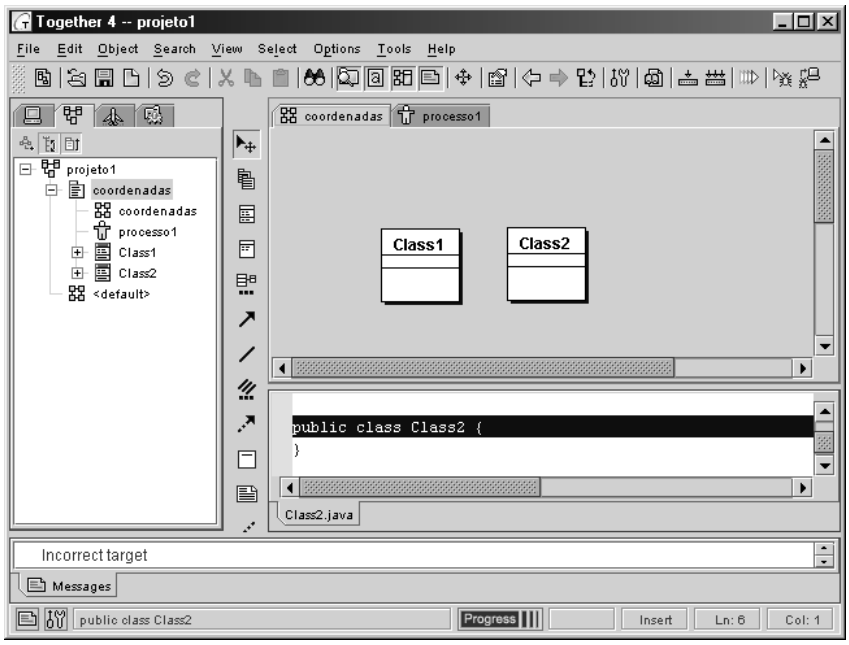
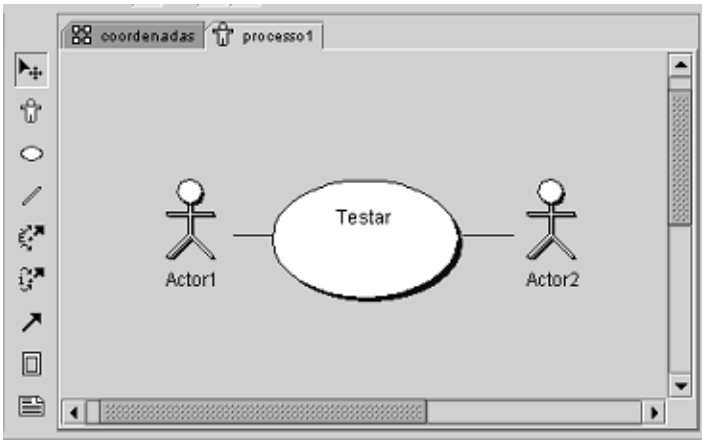


FIGURA 68 - EXEMPLO DE DIAGRAMA *use-case*



4.3 PROGRAMA *Triangle*

O programa *Triangle* é definido pelos seus criadores como sendo um gerador de estruturas triangulares bidimensionais voltadas para simulação de elementos finitos (SHEWCHUK, 2004). Este programa foi desenvolvido pelo esforço conjunto de pesquisadores dos Departamentos de Engenharia Civil, Engenharia Ambiental e

Ciência da Computação da Universidade de *Carnegie Mellon*. Este programa é distribuído livre de taxa e pode ser obtido via Internet no endereço: http://www.cs.berkeley.edu/~jrs/triangle.shar.gz.

Para rodar o *Triangle* é necessário dispor os dados segundo a organização da fig. 69 que tem extensão “node”. Na primeira linha estão representados, respectivamente, o número total de pontos (ou nós) a serem triangulados (288), a especificação do espaço dimensional para gerar a triangulação (2 - bidimensional), o número de atributos extras para cada ponto (1 - significa que além das duas coordenadas planimétricas existe mais um valor que corresponde à altitude) e no último campo é informado se o arquivo contém marcadores de fronteiras (0 = não contém). É importante destacar que o valor do espaço dimensional é obrigatoriamente igual a dois. Nas linhas restantes são apresentados os valores que identificam cada ponto e as suas coordenadas (coordenada Este =677683.671149, Norte =7183800.243055 e Altitude =913.640000, que é o atributo extra mencionado anteriormente). Para se especificar uma linha de comentário é utilizado o caractere “#” antes do comentário.

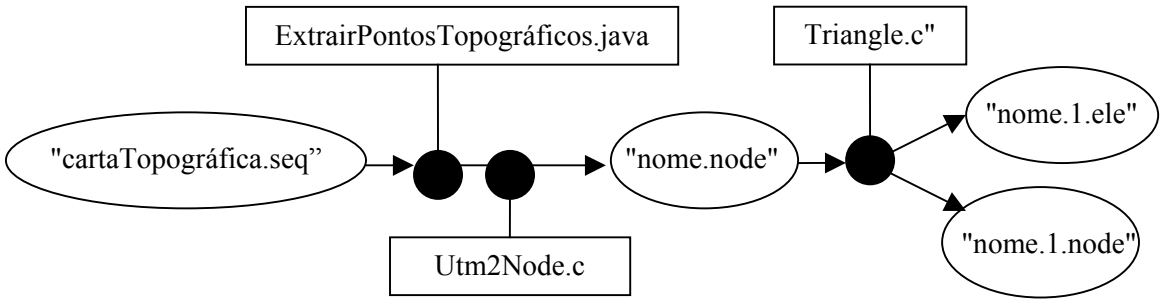
FIGURA 69 - FORMATO DO ARQUIVO ("NODE") PARA O *Triangle*

```
288 2 1 0
1 677683.671149 7183800.243055 913.640000
2 677644.713617 7183755.215083 915.510000
3 677677.499633 7183746.242292 918.620000
4 677722.757416 7183725.670572 914.630000
5 677724.300295 7183771.442648 913.620000
.................
# "nome.node"
```

Para se realizar os experimentos iniciais, concebeu-se um programa em Java que, a partir das cartas topográficas digitais, permitia a extração somente dos pontos necessários para gerar a superfície topográfica (normalmente, as de curvas de níveis e os pontos altimétricos) como sendo uma única lista de pontos com um identificador numérico e três coordenadas. Em seguida, estes dados eram utilizados como entrada para um módulo em linguagem C chamado "Utm2node.c" desenvolvido por Olivier

Mathias van Kaick[5]. Este módulo organiza os dados de entrada para o programa *Triangle* (ver formato “node” na fig. 69). Na fig. 70 é apresentado o fluxo de processamento que se realizava para obter a estrutura de Delaunay.

FIGURA 70 - FLUXO DE PROCESSAMENTO PARA GERAR A TRIANGULAÇÃO DELAUNAY



Na fig. 71 são apresentados os formatos dos arquivos gerados pelo programa *Triangle*. No arquivo com extensão "1.node" (fig. 71a) estão armazenados os vértices e no arquivo com extensão "1.ele" (fig. 71b) estão armazenadas as faces geradas pela triangulação. Os quatros valores apresentados na primeira linha da fig. 71a representam, respectivamente, o número de vértices, a dimensão (o valor dois quer dizer bidimensional), o número de atributos (neste caso só existe um atributo que é a altitude) e o código que identifica se o vértice é um vértice da envolvente (o valor zero quer dizer que o vértice não pertence a envolvente e é interno). Os três valores apresentados na primeira linha da fig. 71b são, respectivamente, o número de faces, a dimensão e o número de atributos. A partir dos dados contidos nestes arquivos, produz-se uma cena VRML utilizando-se o módulo "tri2wrl.java" (fig. 72).

FIGURA 71 - FORMATO DOS ARQUIVOS "1.NODE" E "1.ELE" SAÍDAS DO *Triangle*

a) Arquivo com extensão “1.node”

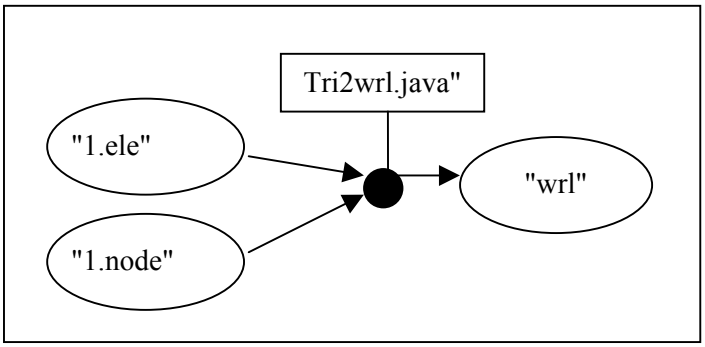
```
288  2  1  1
1 677683.67114899994   7183800.243055        913.63999999999999  0
2 677644.71361700003   7183755.2150830003    915.50999999999999  0
3 677677.499633        7183746.242292        918.62              0
4 677722.75741600001   7183725.6705719996    914.63              0
5 677724.30029499996   7183771.4426480001    913.62              0
6 677746.41000000003   7183800.7599999998    912.82000000000005  0
.....
```

b) Arquivo com extensão “1.ele”

```
549  3  0
1  156   155   151
2  155   156   157
3  19   151    18
4  150   151   155
5  155   149   150
6  156   151   152
.....
```

[5] Aluno do Curso de Infomática da UFPR.

FIGURA 72 - GERAÇÃO DA SUPERFÍCIE TOPOGRÁFICA SEGUNDO O PADRÃO VRML



Deve ser destacado que o programa *Triangle* foi concebido originalmente para rodar em ambiente Unix e sem interface gráfica. Posteriormente então, desenvolveu-se um novo processo em que foram integrados todos os módulos (extração de pontos, conversão para formato *Triangle* e a chamada para executar o programa *Triangle*) num único fluxo de processamento. Atualmente, isto é feito de forma transparente para o usuário e através de uma interface gráfica. Entretanto, é necessário disponibilizar em algum diretório a carta topográfica digital para que esta seja inserida na base de dados. Após isto ter sido feito, pode-se acionar o módulo principal em que se especifica (ou seleciona) um projeto. Em seguida, pode-se selecionar a opção "Importar Levantamento" (fig. 73) que disponibiliza o recurso de navegação através dos diretórios (fig. 74). Assim, pode-se localizar e selecionar uma carta topográfica em que estão contidos os dados.

FIGURA 73 - INTERFACE PARA SELEÇÃO DO ARQUIVO COM O LEVANTAMENTO

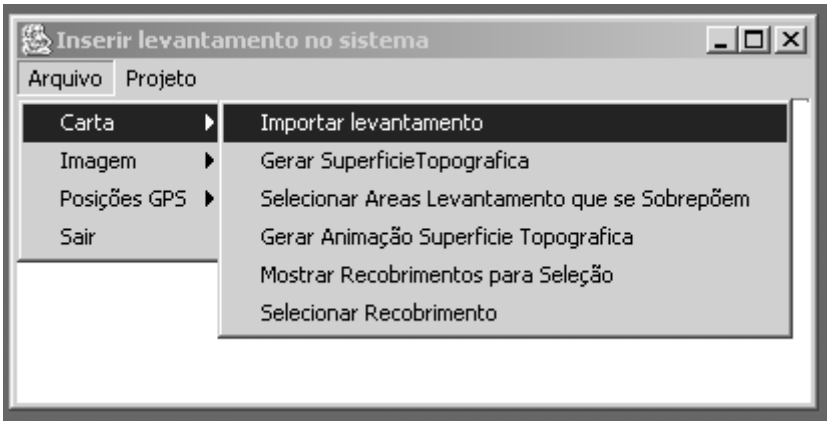
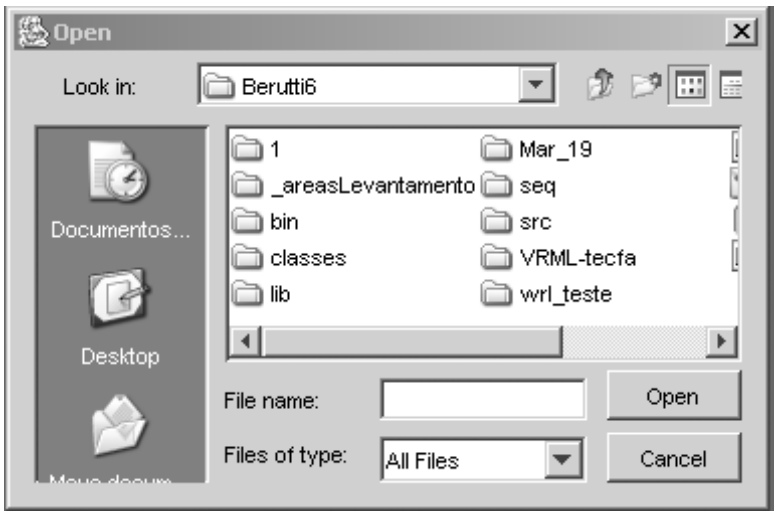


FIGURA 74 - RECURSO DE NAVEGAÇÃO PARA SELECIONAR O ARQUIVO COM OS DADOS DE LEVANTAMENTO



4.4 PROGRAMA MaxiCAD

O programa MaxiCAD é um programa concebido para a criação e edição de cartas topográficas. As informações relativas aos fenômenos terrestres são estruturas como camadas de informação, sendo que cada camada tem um identificador numérico único, uma descrição sucinta do conteúdo da camada de informação e um conjunto de dados estruturados como elementos vetoriais do tipo, pontos, linhas ou áreas. Este programa foi desenvolvido pela empresa MAXIDATA, sediada em Curitiba-PR. Este programa foi utilizado no contexto deste trabalho, simplesmente, para simular dados para a realização de alguns experimentos (ver itens 5.2 e 5.3) e para converter os dados da carta topográfica digital para um formato texto.

4.5 SISTEMA *Cortona*

Cortona foi concebido pela empresa *ParallelGraphics*, com sede na Irlanda. Este sistema é definido pelos seus criadores como sendo um visualizador Web3D interativo (PARALLELGRAPHICS, 2004). Com este é possível interagir e visualizar modelos 3D na *Web*. Uma outra maneira de definir o *Cortona* é caracterizá-lo como sendo um VRML *plug-in*[6] para navegadores de padrão Internet, como por exemplo, o *Internet Explorer* ou o *Netscape Navigator*. Toda vez que um arquivo contendo

[6] *Plug-in* é um programa acessório que adiciona funcionalidade e aparência ao programa principal. São normalmente usados na *Web* em hipertextos.

objetos VRML é acessado com um navegador Internet, o *Cortona* é automaticamente ativado e pode-se visualizar e interagir com esse mundo. O endereço para obter o *Cortona* via Internet é www.parallelgraphics.com.

A janela principal do *Cortona* pode ser dividida em duas partes, que são a janela de visualização do mundo VRML (também chamado de uma cena VRML) e as barras de ferramentas (vertical e horizontal) que ficam em volta da janela de visualização. A barra vertical contém os botões usados para especificar o tipo de navegação e a barra horizontal contém os botões com ações predefinidas para mudar a posição do observador ou do mundo VRML. O movimento do usuário através do mundo VRML é similar ao movimento de uma câmera, que pode ter uma posição e uma orientação independentes. Este conceito tem por base a idéia de que existe um observador hipotético que vê e interage com o mundo VRML. É possível também serem observadas cenas estereoscópicas desde que seja instalado um *hardware* específico e usado um óculos especial para este fim.

4.5.1 Posicionamento e Movimentação Sobre a Cena

Existem situações em que o autor da cena quer guiar o observador da cena, ditando o seu posicionamento e orientação inicial. Para isto, o autor da cena especifica uma lista de pontos de vistas que o observador pode acessar pelo botão de controle chamado ponto de vista (*viewpoint*) sobre a barra horizontal.

Para se movimentar sobre a cena VRML, um observador pode usar o mouse, o teclado ou ambos. No que se refere aos modos de movimentação, existem três alternativas, que são *Walk*, *Fly*, e *Study*. Para selecioná-las é só clicar com o botão direito do mouse sobre o respectivo botão de controle na barra vertical. Além destes modos de navegação, existem quatro opções alternativas (*Plan, Pan, turn*, e *Roll)* que são usadas de forma combinada com os modos de navegação e que determinam as possibilidades de movimentos e orientações da câmera. Algumas dessas combinações foram testadas e têm a seguinte função:

a) *Walk* + *Plan* – caracteriza um movimento sobre um plano horizontal e permite que o observador se aproxime ou se afaste da cena;

b) *Walk + Pan* – caracteriza um movimento sobre um plano horizontal e permite que o observador se desloque à esquerda ou à direita;
c) *Walk + Turn* – caracteriza um movimento de rotação sobre o eixo horizontal ou sobre o eixo vertical;

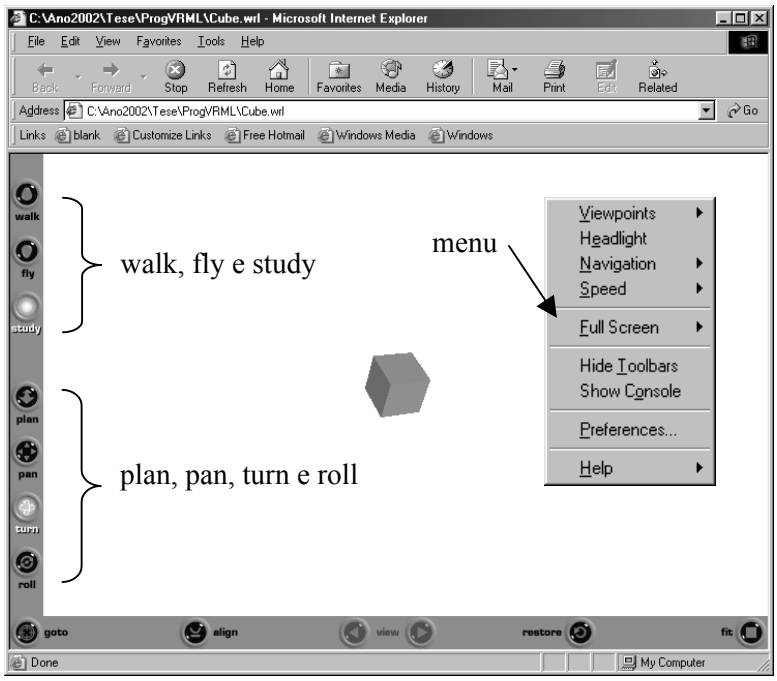
Na fig. 75 é apresentada a janela principal do *Cortona* e são destacadas as localizações dos modos de movimentação e as opções alternativas. Além disto, é apresentado também o menu a partir do qual se pode inicializar algumas das variáveis. Estas variáveis são utilizadas para especificar a interface do usuário, o modo de navegação e as propriedades ambientais que serão usadas no *Cortona*. Para isto, existe um conjunto de opções que são acessadas pelo menu (*pop-up menu*). Para visualizar este menu, deve-se clicar com o botão direito do mouse sobre a janela de visualização (fig. 75). Dentre as opções possíveis deste menu tem-se:

a) pontos de vista (*Viewpoints*) - permite que o usuário selecione distintos pontos de vista, desde que o autor da cena tenha especificado;
b) iluminação principal (*Headlight*) - permite que o usuário ligue ou desligue uma fonte de luz na cena VRML. Este tipo de fonte está posicionado de modo a iluminar frontalmente a cena VRML;
c) navegação (*Navegation*) - permite ao usuário selecionar o modo de navegação (*walk*, *fly* e *study*);
d) velocidade (*speed*) - permite ao usuário controlar a velocidade de navegação sobre a cena (variando entre lenta, normal e rápida);
e) tela (*Full screen*) - permite ao usuário mudar o tamanho da janela de visualização;
f) esconder as barras de ferramenta (*Hide toolbars*) - permite ao usuário esconder ou apresentar as barras de ferramenta;
g) mostrar o console (*Show console*) - permite ao usuário mostrar ou esconder o console usado para comunicação com o usuário;

h) tipos de preferências (*Preference*) - permite ao usuário modificar os valores originais das propriedades relativas à aparência do ambiente e efeitos de renderização (efeitos que dão a impressão de realidade);

i) ajuda (*Help*) - permite ao usuário acessar o sistema de ajuda e informações sobre o *Cortona*.

FIGURA 75 - JANELA PRINCIPAL DO *Cortona* E SEUS BOTÕES DE CONTROLE



4.6 MICROCOMPUTADOR

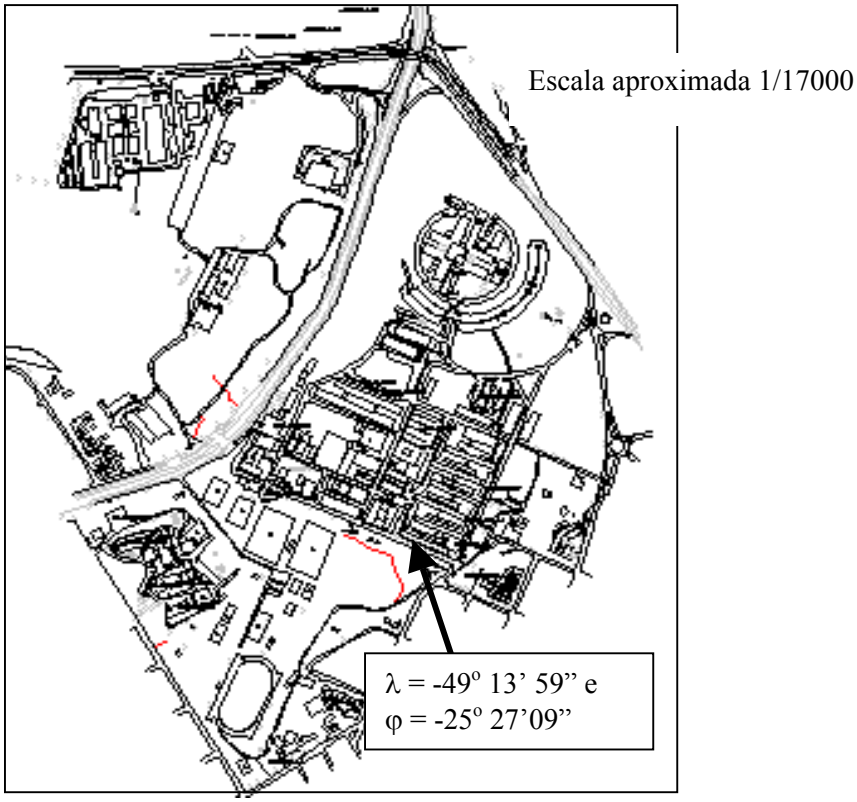
Para o processamento dos dados foi utilizado um micro computador com a seguinte configuração: processador AMD de 1700MHz; disco rígido de 40GB; 512MB de memória *RAM*; e sistema operacional Windows 2000, recursos que estão alocados no Laboratório de Cartografia e GIS do Curso de Pós-Graduação em Ciências Geodésicas da UFPR.

4.7 DADOS

Para a realização dos experimentos, utilizou-se como base para a simulação dos dados uma carta topográfica digital 3D, escala 1/2000, produzida em julho de

2001 por uma equipe de professores do Departamento de Geomática da Universidade Federal do Paraná (DEPARTAMENTO DE GEOMÁTICA, 2001). Na fig. 77 é apresentada esta carta em que estão sendo visualizadas as edificações e os arruamentos do Centro Politécnico da UFPR, com área aproximada de 5500m^2. As coordenadas longitude e latitude, do Bloco VI, onde se encontram o Departamento de Geomática e o Curso de Pós-Graduação em Ciências geodésicas, são, respectivamete, λ = -49^o 13' 59" e φ = -25^o 27'09".

FIGURA 76 - CARTA TOPOGRÁFICA DIGITAL DO CENTRO POLITÉCNICO



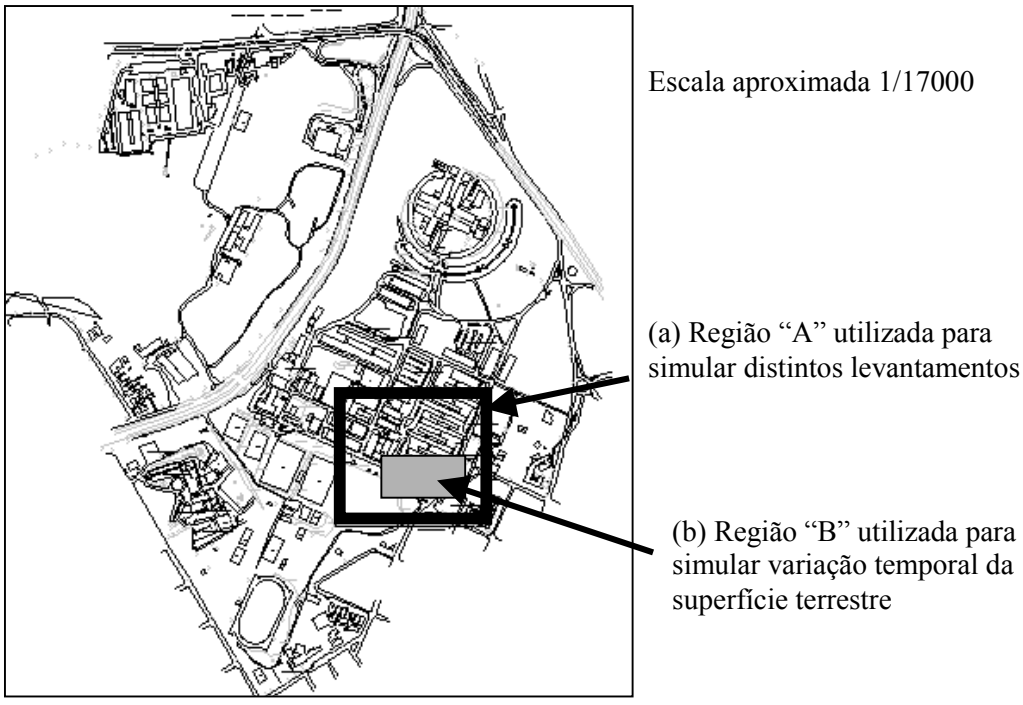
5 EXPERIMENTOS E RESULTADOS

O principal resultado obtido com esta Tese é de caráter conceitual e diz respeito ao modelo proposto para representar fenômenos terrestres que têm uma semântica definida e que podem ser caracterizados espacialmente por uma superfície. Para demonstrar isto foram realizados experimentos que tiveram por base os dados oriundos de uma carta topográfica digital do Centro Politécnico.

5.1 CARTA TOPOGRÁFICA DIGITAL DO CENTRO POLITÉCNICO

Como foi descrito no item 4.7 os dados utilizados nos experimentos tiveram por base a carta topográfica digital 3D do Centro Politécnico, escala 1/2000. Para se analisar, detalhadamente, as funcionalidades esperadas para os métodos implementados foram definidas outras duas regiões menores, chamadas de regiões "A" e "B" (fig. 77). Enquanto a região "A" foi usada para estabelecer os métodos que caracterizam as relações espaciais entre áreas de levantamento, a região "B" foi usada para determinar a variação temporal e simular dados para obter a variação temporal para mais três épocas.

FIGURA 77 - REGIÃO "A" E REGIÃO "B"



Durante os experimentos com os dados da carta digital, observou-se que algumas feições (por exemplo, algumas linhas de meio fio, linhas de delimitação de canteiros, cantos de edificações e elementos pontuais como postes de iluminação) apresentavam valores altimétricos inconsistentes quando comparados com as curvas de nível mais próximas. Além disto, para os elementos do tipo "árvore" a informação altimétrica disponível era relativa apenas ao topo da árvore. Em função disto, para se gerar a superfície topográfica somente foram utilizados os valores altimétricos que tinham origem a partir de curvas de nível (mestra e intermediária) ou de pontos altimétricos sobre a superfície topográfica. Na fig. 78 é apresentada a estrutura de Delaunay resultante para um conjunto de 13120 pontos e na fig. 79 a correspondente imagem segundo o padrão VRML é apresentada segundo uma vista isogonal.

FIGURA 78 - ESTRUTURA DE DELAUNAY PARA OS DADOS DO CENTRO POLITÉCNICO

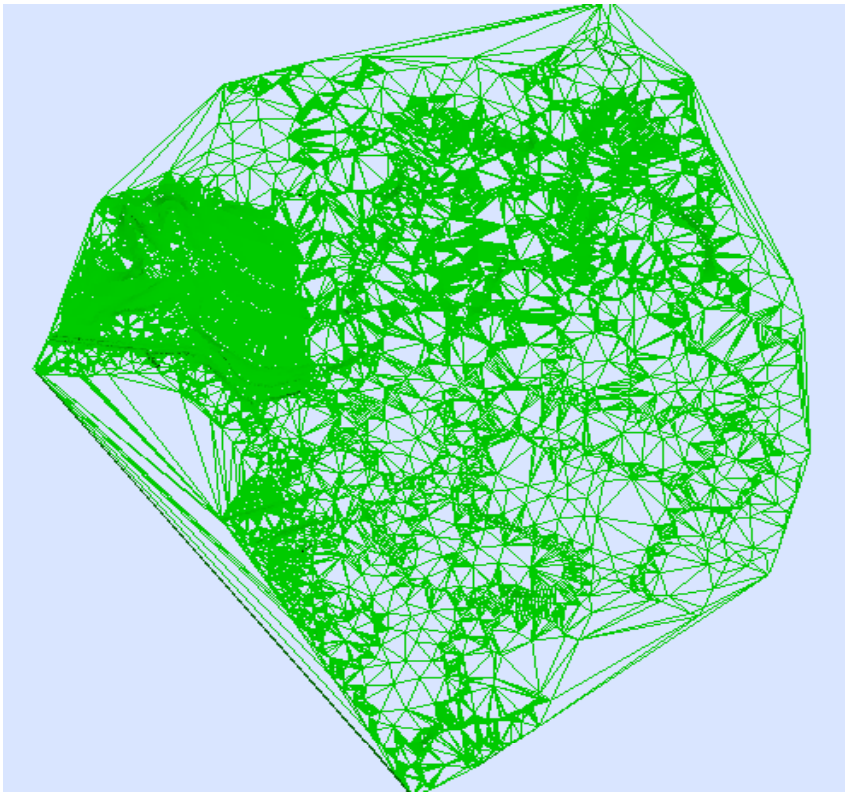
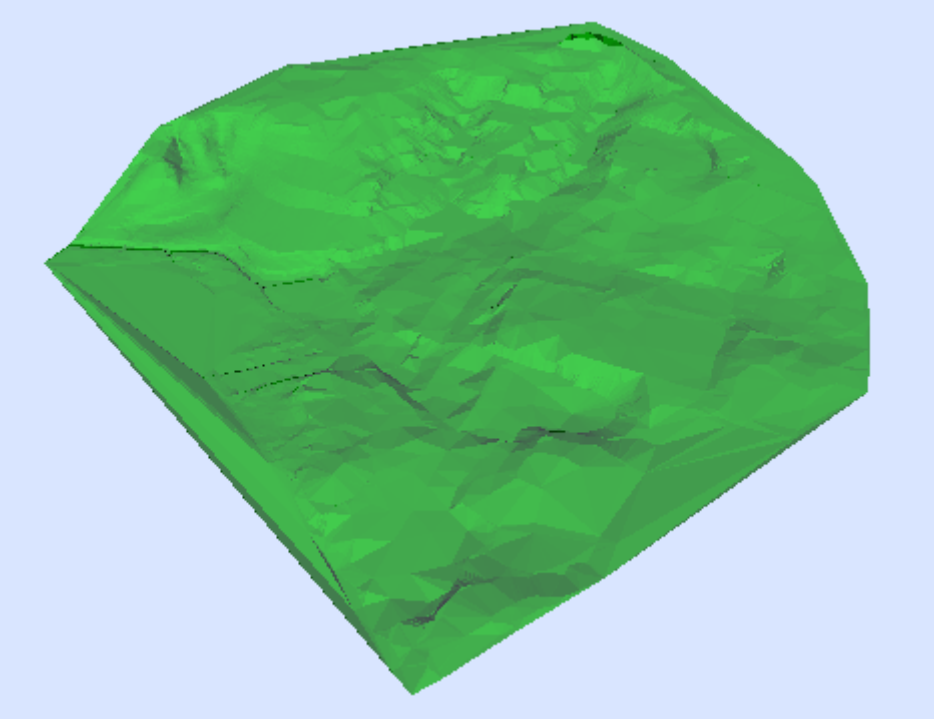


FIGURA 79 - SUPERFÍCIE TOPOGRÁFICA PARA O CENTRO POLITÉCNICO



5.1.1 Região “A”

Esta região foi recortada da carta digital e está representada na fig. 80. Após ter sido editada (para garantir que não houvesse inconsistência quanto à existência de pontos coincidentes e altitudes erradas) foi realizada a triangulação de Delaunay e gerada a superfície topográfica segundo o padrão VRML (fig. 81). Além disto, fez-se um particionamento desta região em sete sub-regiões que foram usadas como base para simular diferentes levantamentos e os possíveis relacionamentos espaciais entre estes (fig. 82).

FIGURA 80 - REGIÃO “A” RECORTADA DA CARTA DIGITAL E EDITADA

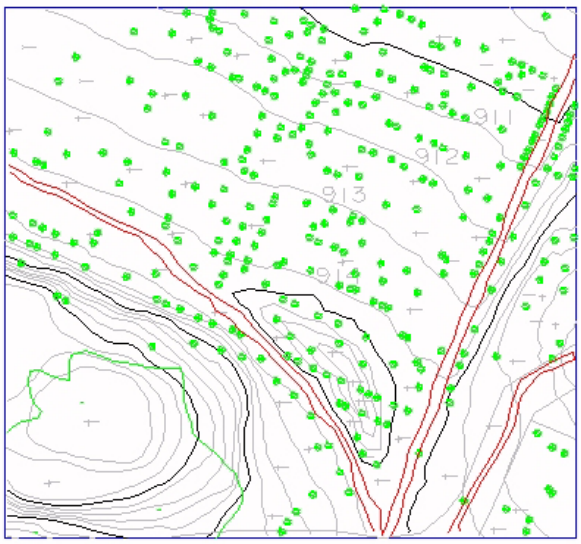


FIGURA 81 - SUPERFÍCIE TOPOGRÁFICA PARA A REGIÃO “A”

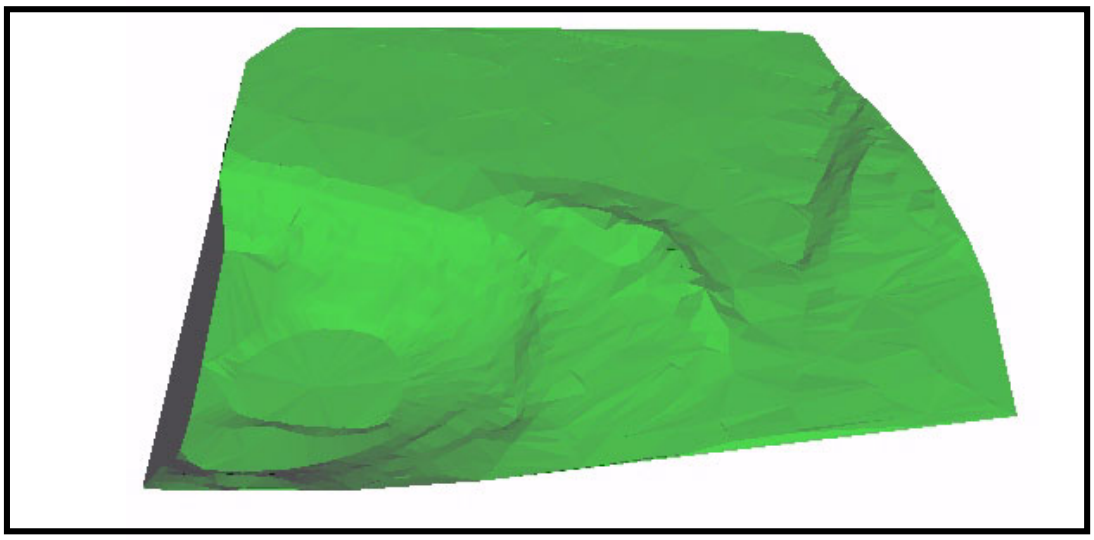
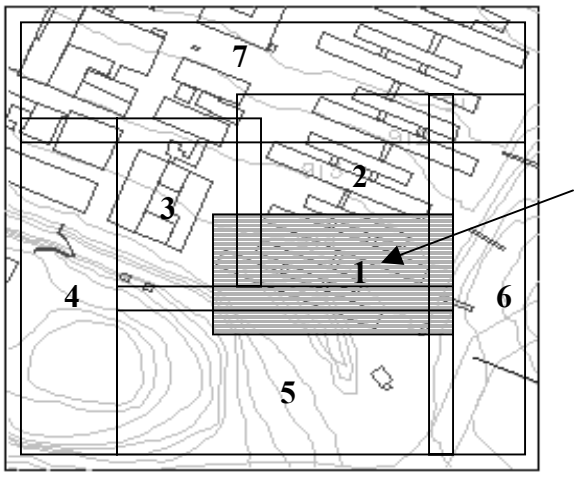


FIGURA 82 - PARTICIONAMENTO DA REGIÃO “A” EM SETE SUB-REGIÕES



Região utilizada para simular a variação temporal da superfície terrestre

5.1.2 Região “B”

Para realizar os experimentos visando à determinação da variação temporal, extraiu-se da região “A” (fig. 82 apresentada anteriormente) um fragmento que chamou-se de região “B” (fig. 83), que considerou-se como relativa a época zero. A partir desta região foram simuladas mudanças sobre a superfície topográfica para outras três épocas que são mostradas nas figs. 84, 85 e 86, representando, respectivamente, as épocas um, dois e três.

FIGURA 83 - SIMULAÇÃO DA ÉPOCA ZERO

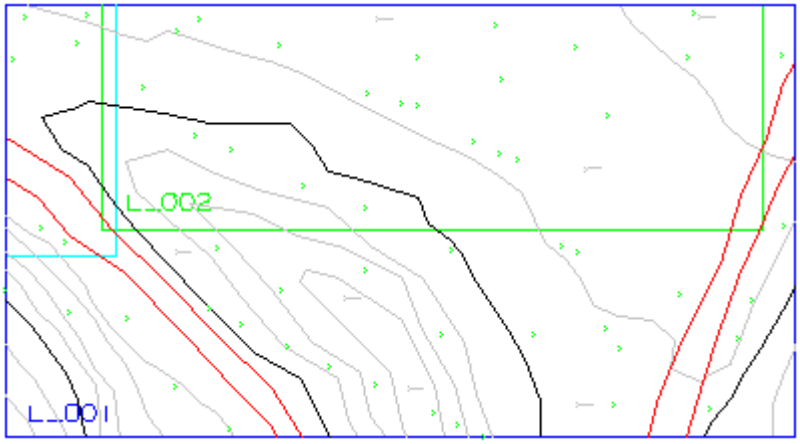


FIGURA 84 - SIMULACÃO DA ÉPOCA UM

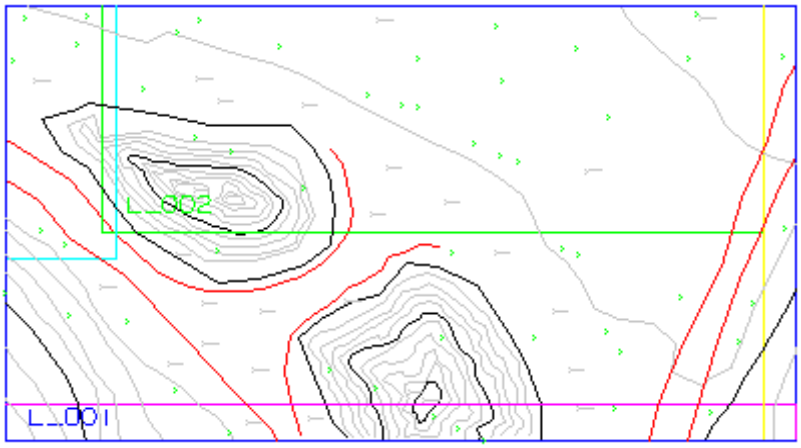


FIGURA 85 - SIMULAÇÃO DA ÉPOCA DOIS

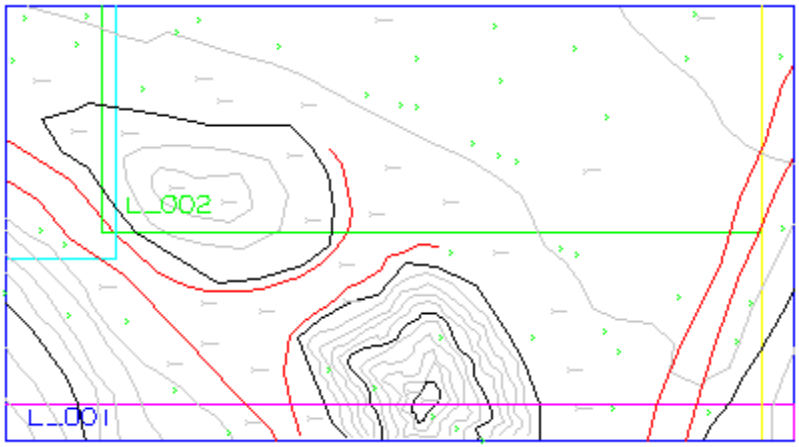


FIGURA 86 - LEVANTAMENTO QUE CARACTERIZA A ÚLTIMA ÉPOCA

Nas figuras 87, 88, 89 e 90 são apresentadas as superfícies topográficas, segundo o padrão VRML, para as quatro épocas relativas respectivamente às figuras 83, 84, 85 e 86. Pode-se observar que essas superfícies não apresentam formas idênticas uma vez que são constituídas por distintos conjuntos de pontos.

FIGURA 87 - SUPERFÍCIE TOPOGRÁFICA PARA A ÉPOCA ZERO

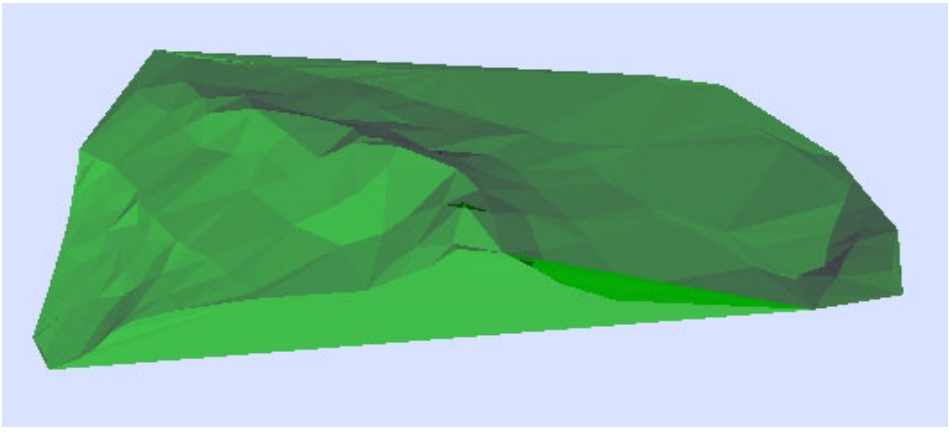


FIGURA 88 - SUPERFÍCIE TOPOGRÁFICA PARA A ÉPOCA UM

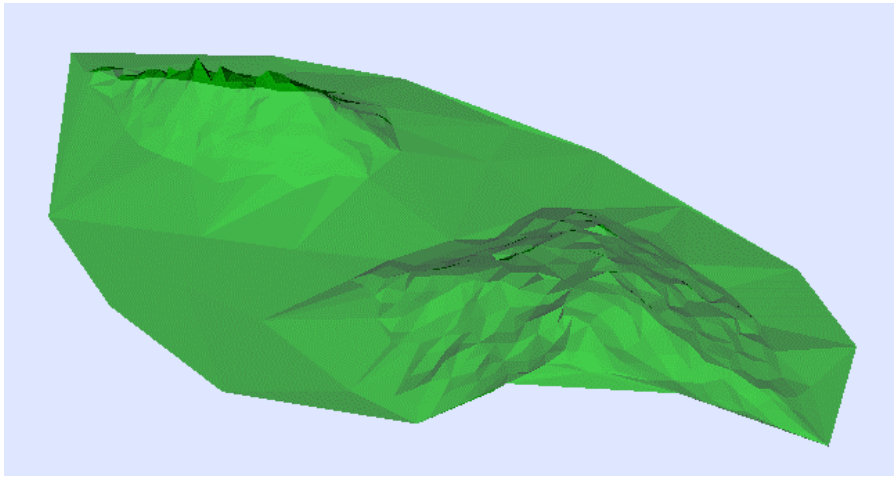


FIGURA 89 - SUPERFÍCIE TOPOGRÁFICA PARA A ÉPOCA DOIS

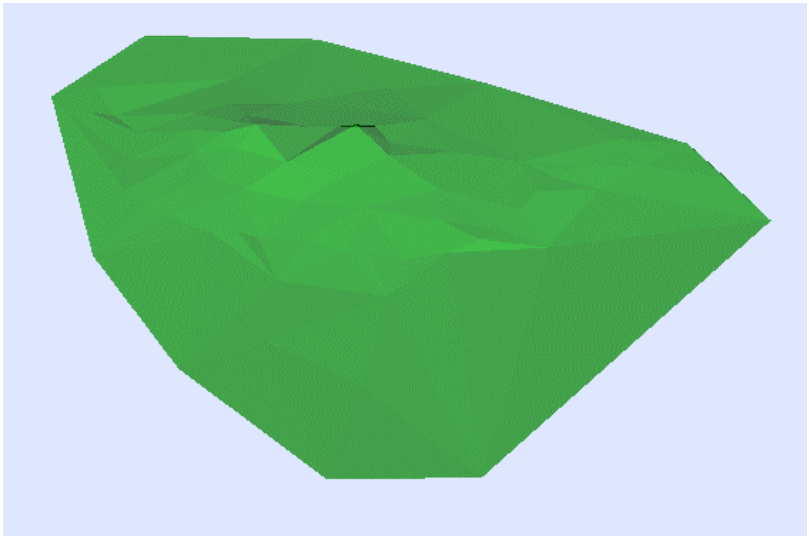
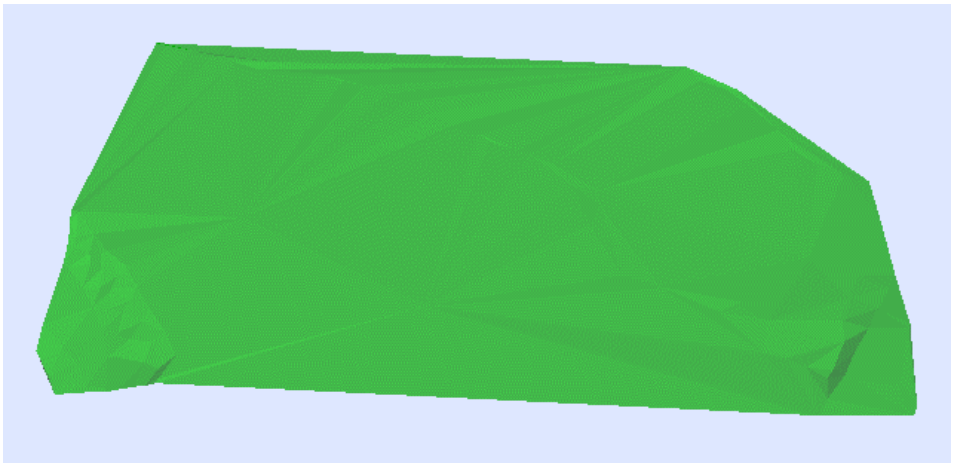


FIGURA 90 - SUPERFÍCIE TOPOGRÁFICA PARA A ÉPOCA TRÊS



5.1.3 Obtenção da Variação Temporal

A seguir são apresentados os resultados obtidos para a variação temporal da superfície topográfica, que consiste na interpolação das altitudes para cada superfície topográfica nas distintas épocas (fig. 91, fig. 92, fig. 93 e fig. 94). Pode-se observar que as superfícies geradas têm a mesma forma porque são constituídas pelo mesmo conjunto de pontos. Para se visualizar os resultados as superfícies geradas para cada época são organizadas como protótipos externos segundo o padrão VRML (ver item 2.7.3 Prototipação).

FIGURA 91 - PROCESSAMENTO PARA A ÉPOCA ZERO

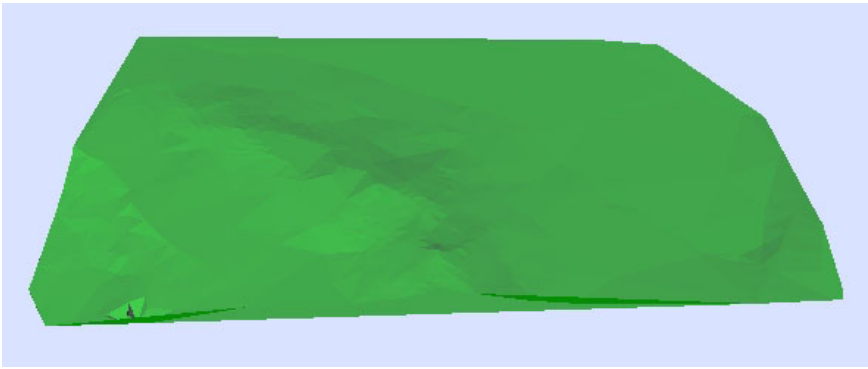


FIGURA 92 - PROCESSAMENTO PARA A ÉPOCA UM

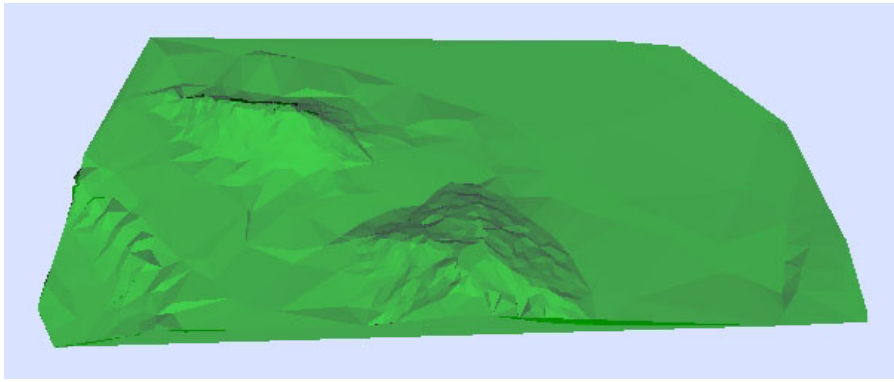


FIGURA 93 - PROCESSAMENTO PARA A ÉPOCA DOIS

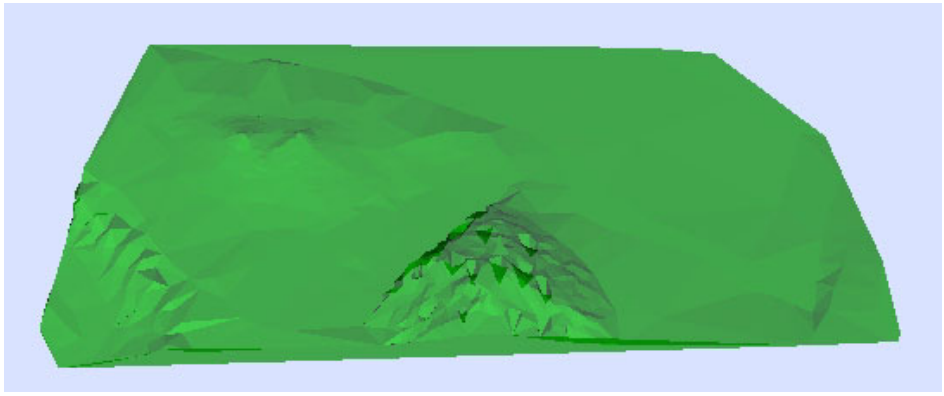


FIGURA 94 - PROCESSAMENTO PARA A ÉPOCA TRÊS

Durante esses experimentos, percebeu-se que as classes implementadas pelo Grupo GeoVRML nem sempre apresentam uma equivalência total com as suas correspondentes classes no padrão VRML. Por exemplo, segundo o padrão VRML existem cinco tipos de interpoladores, que são: **CoordinateInterpolator**; **OrientationInterpolator**; **NormalInterpolator**; **ColorInterpolator; PositionInterpolator;** e **ScalarInterpolator** (ver item 2.7.4.2 Nós do tipo interpolador). Entretanto, no pacote GeoVRML só existe o **GeoPositionInterpolator**. Em função disto, não pode-se visualizar uma variação temporal usando o pacote GeoVRML. Conseqüentemente, foi necessário alterar o código original da classe inserindo um manipulador de eventos capaz de tratar a interpolação das coordenadas.

5.1.4 Outros Resultados

Considerando que para o levantamento da época três estavam disponíveis dados relativos aos fenômenos terrestres do tipo árvore, luminária e edificação, implementou-se as correspondentes classes chamadas respectivamente de “Arvore”,

“Luminaria” e “Edificacao” (fig. 95). O objeto “Luminaria” e o objeto “Arvore” foram considerados como de dimensionalidade 0D porque, para estes somente eram disponíveis as suas posições. Para a apresentação destes foram criados protótipos segundo o padrão VRML que são apresentados nas fig. 96 e fig. 97. Adicionalmente, foi utilizado um padrão de textura para recobrir o objeto “Arvore” que é composto por dois tipos básicos de objetos VRML: “Cylinder” e “Cone”. Para instanciar o objeto “Edificação” foi necessário conceber um método que constrói as faces da edificação com base nos vértices da planta baixa e da altura da edificação.

FIGURA 95 - VISUALIZAÇÃO DE OBJETOS “Edificacao”, “Arvore” E “Luminaria”

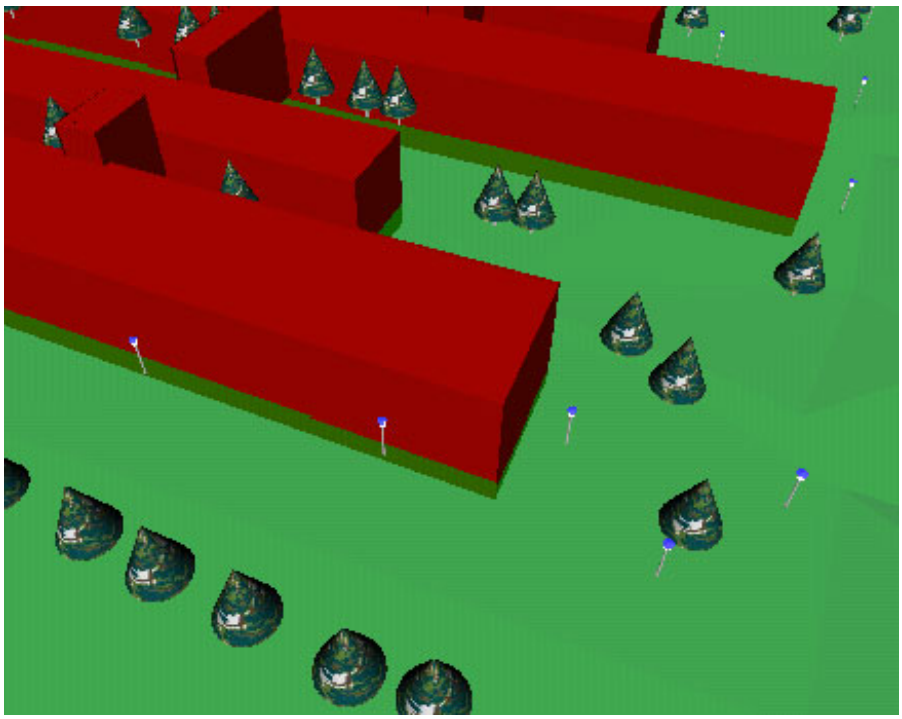


FIGURA 96 - COMPOSIÇÃO DO OBJETO “Luminaria”

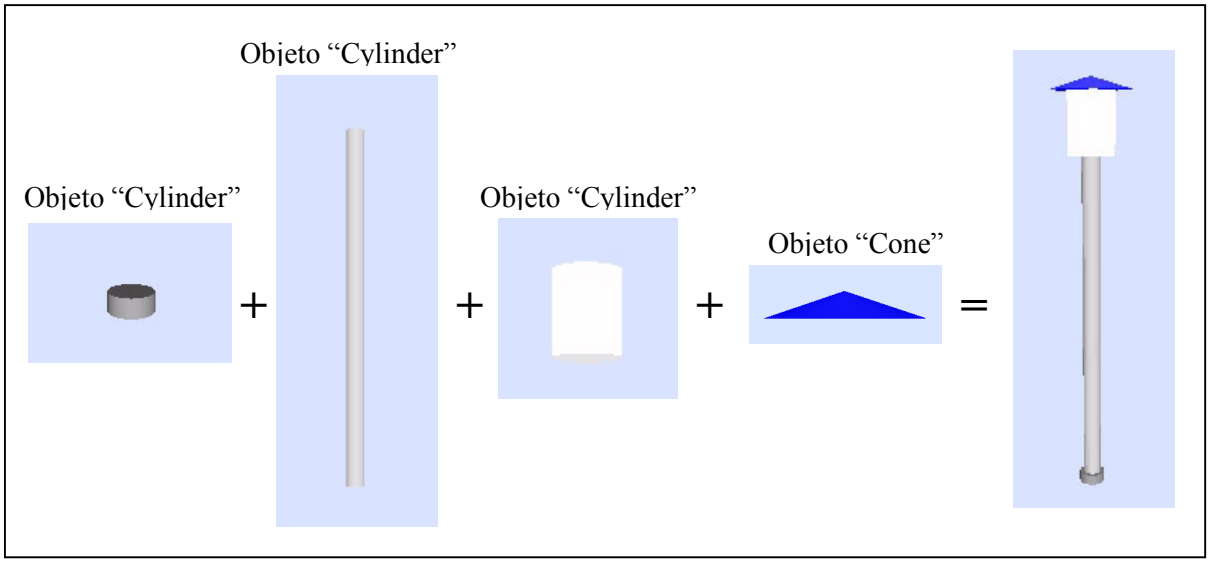
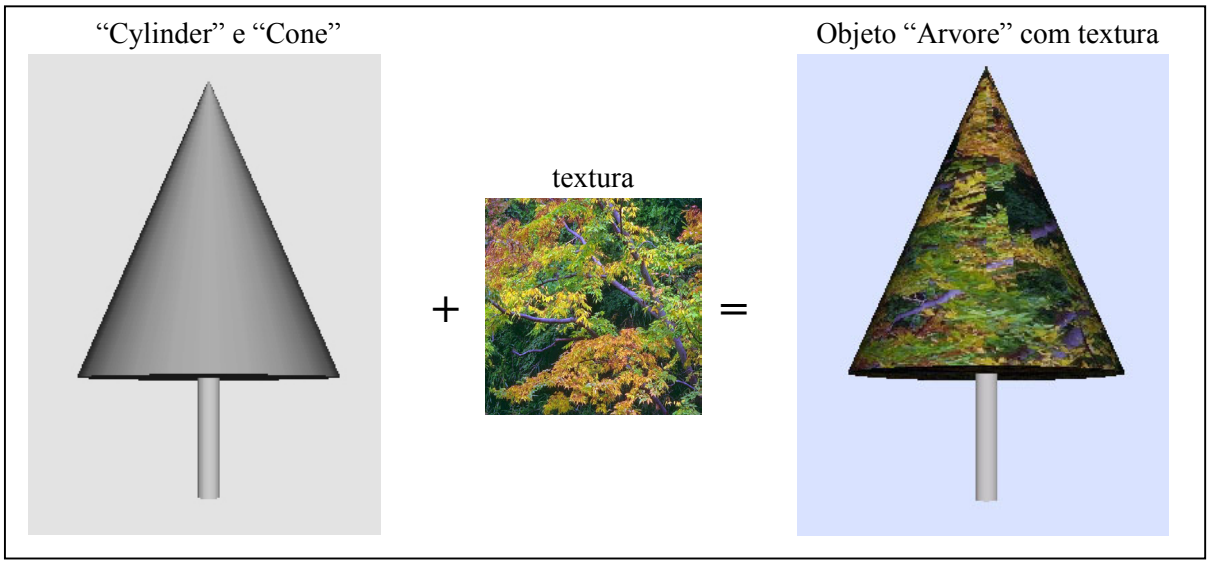


FIGURA 97 - COMPOSIÇÃO DO OBJETO “Arvore”



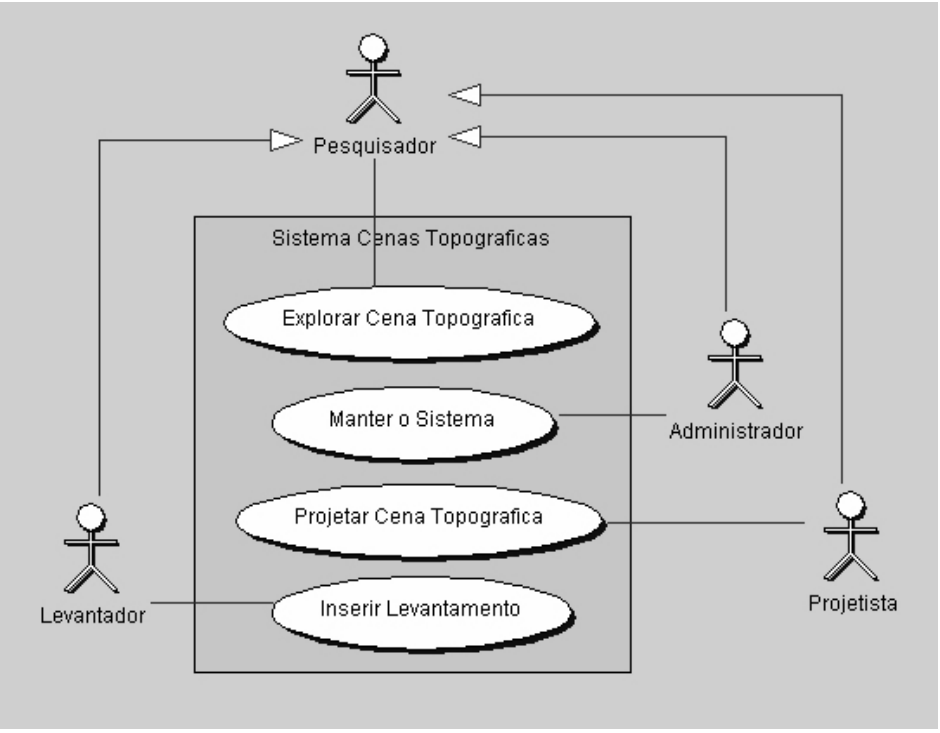
5.2 MODELANDO SEGUNDO O PADRÃO UML

A seguir são apresentados os resultados da modelagem segundo o padrão UML gerados com o programa *Together*. Nestes destacam-se os tipos de atores envolvidos e os principais pacotes, que são: “persistência”; “visualização”; “fenomenoTerrestre”; “areaLevantamento”; e “recobrimento”. Como o modelo proposto está voltado para a representação de fenômenos terrestres sob o ponto de vista da cartografia topográfica, denominou-se o sistema como sendo “Sistema para Fenômenos Terrestres”.

5.2.1 Tipos de Usuários

A partir da modelagem foram identificados quatro tipos de usuários. O primeiro é o usuário pesquisador. Este usuário é o mais genérico e interage com o sistema por meio do "use-case" denominado de "Explorar Cena" que consiste em selecionar um projeto dentre os disponíveis e interagir com os fenômenos da cena(s). Para um mesmo projeto pode existir uma ou mais cenas. Os tipos de iteração previstos são selecionar, visualizar e gravar cenas ou fenômenos segundo o padrão VRML. Além disto, este usuário pode também simular novos fenômenos numa cena. Um outro usuário é o administrador, que é responsável pela manutenção e integridade operacional do sistema. Este usuário especializa o usuário pesquisador e tem a função de eliminar objetos do tipo "Projeto", "Cena", "AreaLevantamento" e "FenomenoTerrestre". Um outro usuário é o levantador, que também especializa o usuário pesquisador e é responsável pela obtenção dos levantamentos e inserção no sistema. O último usuário como previsto é o projetista, que especializa o usuário pesquisador e é responsável por criar/alterar objetos ("Projeto", "Cena", AreaLevantamento" e "Fenomenoterrestre"), disponibilizar objetos ("Projeto", "Cena" e "Fenomenoterrestre") e solicitar a eliminação de objetos (fig. 98).

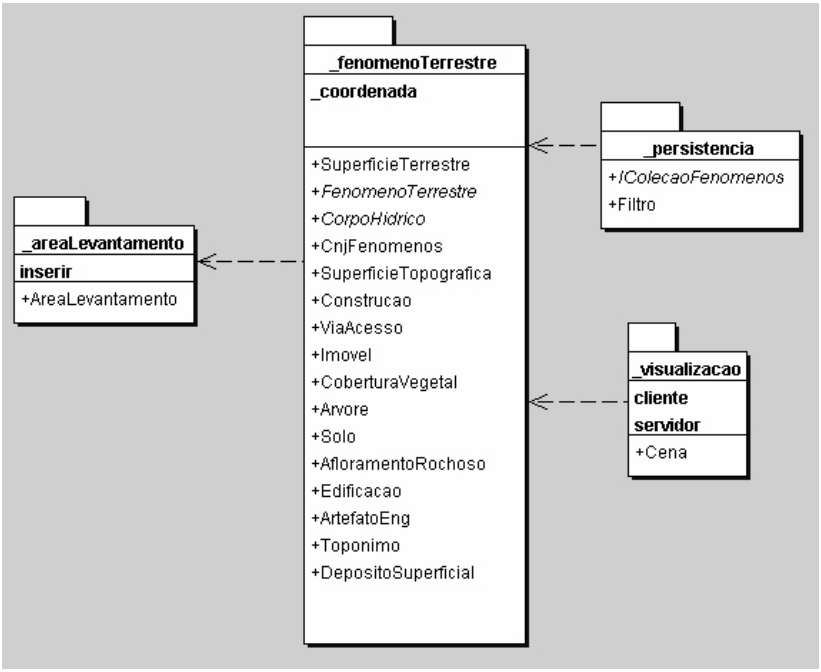
FIGURA 98 - MODELAGEM DOS TIPOS DE USUÁRIOS COM UML



5.2.2 Principais Pacotes

Os principais pacotes modelados foram: "_persistência"; "_visualização"; "_fenômenoTerrestre"; "_areaLevantamento"; e "_recobrimento". Nestes pacotes estão agrupadas as classes que são afins. Na fig. 99 é apresentada a modelagem destes quatro pacotes.

FIGURA 99 - MODELAGEM DOS PRINCIPAIS PACOTES



5.2.3 Fenômenos terrestres

A partir dos objetos "AreaLevantamento" são instanciados então os objetos do tipo "FenomenoTerrestre". Para o armazenamento e recuperação é prevista a utilização do pacote "_persistencia" (LIMA, 2004) e para a visualização é prevista a utilização do pacote "_visualizacao" (CAMARGO, 2003). Entretanto, como o pacote de persistência está em desenvolvimento e a integração destes exigirá um esforço adicional, alternativamente não fez-se persistência dos objetos instanciados e utilizou-se o padrão VRML para visualizar os fenômenos terrestres. Na fig. 100 são apresentados os tipos de classes de fenômenos terrestres que foram modeladas.

5.2.4 Pacote "_coordenada"

Para lidar com a forma de referenciamento espacial dos objetos foi criado o pacote "_coordenada", em que estão agrupadas as classes que fazem referência aos sistemas coordenadas geodésica (latitude, longitude e altitude), geocêntrica (X, Y, Z) e UTM (norte, este e altitude) como implementados e disponíveis no pacote de classes GeoVRML. Na fig. 101 é apresentada a modelagem dessas classes.

FIGURA 100 - MODELAGEM DOS FENÔMENOS TERRESTRES

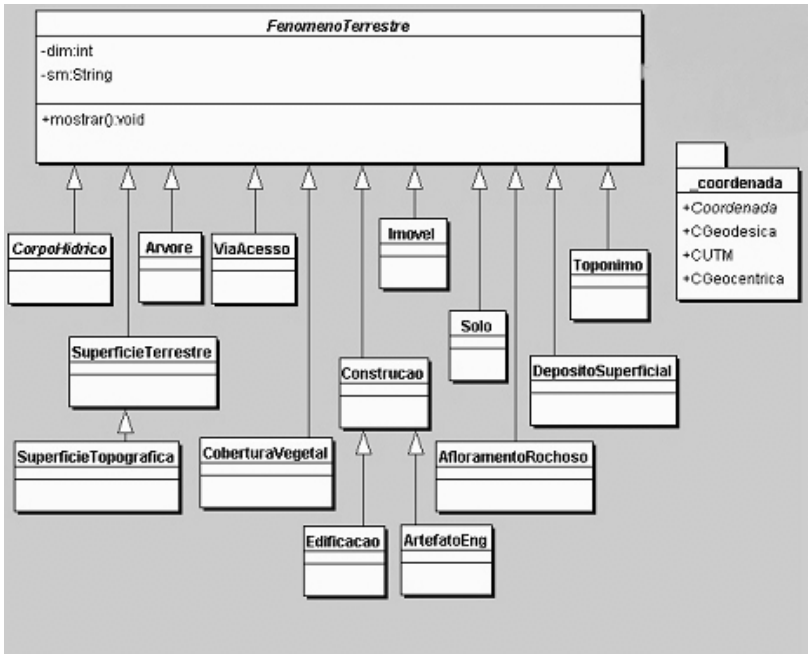
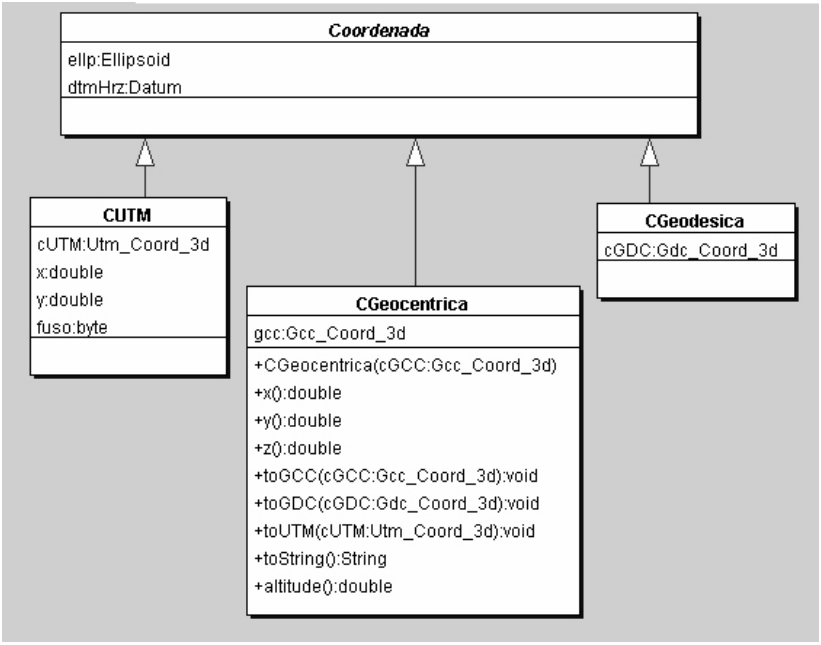


FIGURA 101 - PACOTE "_coordenada"



6 CONCLUSÕES E RECOMENDAÇÕES

6.1 MODELO PROPOSTO

O modelo proposto neste trabalho é destinado à representação de fenômenos terrestres, que são apresentados em cartas topográficas. Para modelar tais fenômenos, considerou-se que a superfície terrestre é o primeiro de todos os fenômenos e sobre esta superfície repousam os outros fenômenos. Cada fenômeno terrestre tem uma semântica definida e é caracterizado espacialmente pela sua superfície externa. O modelo proposto é classificado como espaço-temporal. Além de serem representadas as características semânticas e as espaciais tridimensionais, podem ser determinadas as variações temporais ocorridas sobre os fenômenos.

Com base nos experimentos realizados ficou demonstrado que o modelo desenvolvido é adequado e funciona segundo os objetivos propostos. Os fenômenos terrestres podem ser representados pelo conjunto de classes modeladas. A variação temporal que um fenômeno terrestre pode apresentar é determinada a partir dos dados de levantamentos realizados em épocas distintas. Tais conclusões são garantidas pelos fatos observados com o desenvolvimento do modelo proposto e nos resultados obtidos a partir dos experimentos realizados, aqui resumidos como:

- o modelo proposto é constituído por três classes principais que são: “FenomenoTerrestre”, “AreaLevantamento” e “Recobrimento”. A classe “FenomenoTerrestre” é uma classe abstrata e geral e possui três membros que são herdados pelas classes que a especializam. O membro “semantica” identifica a classe a qual o fenômeno pertence. O membro “superfície” caracteriza espacialmente o fenômeno, sendo que esta superfície pode ter dimensionalidade: 0D, 1D, 2D ou 3D. O membro “retanguloEnvolvente” define o retângulo que envolve o objeto instanciado;
- para contemplar os tipos de superfície que um fenômeno pode ter, a classe “Superficie” é especializada pelas classes “Superficie0D”,

"Superficie1D", "Superficie2D" e "Superficie3D". Um fenômeno com dimensionalidade igual a zero tem uma superfície que se degenera num único objeto do tipo "Ponto". Um fenômeno com dimensionalidade igual a um tem uma superfície que se degenera numa seqüência ordenada de objetos do tipo "Ponto". Um fenômeno com dimensionalidade igual a dois tem uma superfície caracterizada por um único objeto do tipo "Face". Um objeto com dimensionalidade igual a três tem uma superfície caracterizada por um conjunto de objetos do tipo "Face";

- os tipos de "semantica" que um objeto da classe "*FenomenoTerrestre*" pode assumir estão definidos na tab. 6.1. Este conjunto de classes foi proposto e foi denominado de classes essenciais, porque a partir destas, ou da sua combinação, são definidas as outras classes. Todo o conjunto de classes essenciais foi modelado e representado segundo o padrão UML, mas para a realização dos experimentos somente foram implementadas as classes "SuperficieTerrestre", "SuperficieTopografica", "Edificacao", "Luminaria" e "Arvore", deixando-se para trabalhos futuros a implementação das outras classes;

TABELA 4 - CONJUNTO DE CLASSES ESSENCIAIS

Classe	Nome da classe
Superfície Terrestre	SuperficieTerrestre
Superfície Topográfica	SuperficieTopografica
Corpo hídrico	*CorpoHidrico*
Construção:	
Edificação	Edificacao
Artefato de Engenharia	ArtefatoEngenharia
Via de acesso	ViaAcesso
Imóvel	Imovel
Afloramento Rochoso	AfloramentoRochoso
Solo	Solo
Depósito Superficial	DepositoSuperficial
Cobertura vegetal	CoberturaVegetal
Árvore	Arvore

- a classe "AreaLevantamento foi concebida para acondicionar os objetos do tipo "FenomenoTerrestre". Com esta abordagem foi possível conceber e implementar os modelos (representação, persistência e visualização) de

maneira modular e independente entre si, porque a interação entre estes passou a se dar pela passagem de objetos do tipo “AreaLevantamento”. Além disto, a classe “AreaLevantamento” tem a função de reter a informação relativa ao instante de observação do fenômeno e o valor da escala para a qual foi feito o levantamento. Um objeto “AreaLevantamento” só pode ser instanciado a partir de um levantamento, sendo portanto esta uma relação um por um;

- a classe “Recobrimento” foi concebida para representar a variação temporal ocorrida sobre os fenômenos, que é determinada a partir dos relacionamentos espaciais entre objetos do tipo “AreaLevantamento”. Para expressar o relacionamento espacial entre “AreaLevantamento” foi identificada uma família de polígonos constituída por: polígonos básicos (“poligonoUniao”, “poligonoIntersecao” e “poligonoBuraco”) e polígonos secundários (“poligonoComplementar”, “poligonoVazio” e “poligonoEnvolvente”. Com os métodos que foram implementados para a classe “Poligono” (“disjunto”, “toca”, “adjacente”, “igual”, “intercepta”, “coincide”, “contém” e “está contido”) é possível delimitar e extrair todo o conhecimento relativo às diferentes épocas. Além disto, pode-se identificar também aquelas áreas que não apresentam variação temporal (um único levantamento) ou então não foram levantadas;
- a variação temporal de um fenômeno terrestre é determinada utilizando-se as áreas delimitadas por “poligonoIntersecao” e “poligonoComplementar”, cuja temporalidade é maior ou igual a um, ou seja, áreas para as quais existem dois ou mais levantamentos. Para estas áreas pesquisa-se e seleciona-se todos os objetos disponíveis para cada época e, então, determina-se a variação temporal comparando-se os dados atribuídos aos correspondentes atributos de cada objeto. Para os experimentos realizados sobre a determinação da variação temporal foi utilizado o objeto “SuperfícieTopográfica”;
- a superfície topográfica é caracterizada pela estrutura de faces e conjunto

de vértices que são produzidos após a triangulação de Delaunay, que é realizada para o conjunto de objetos do tipo "Ponto" cujo membro "tipoTopografico" tem um valor igual a "True", ou seja, são todos os pontos que pertencem à superfície topográfica;

- a linguagem de modelagem UML garantiu o formalismo para expressar o modelo proposto e permitiu uma comunicação e uma compreensão em alto nível das idéias e funcionalidades previstas para as classes. Além disto, agora é possível realizar novas formulações e reformulações das classes existentes e dos tipos de relacionamentos modelados. Isto representa um elemento conceitual importante na evolução do modelo proposto;
- a utilização da linguagem ***Java*** foi importante para se garantir independência de sistema operacional e possibilitar que em trabalhos futuros, estendam-se as funcionalidades do modelo proposto principalmente com vista ao ambiente Web;
- pode-se dizer que a utilização do padrão VRML e do conjunto de classes GeoVRML foi também uma decisão acertada, não somente porque viabilizou os estudos sobre o modelo proposto, mas também porque abriu uma perspectiva para trabalhos futuros em que estejam envolvidos visualização e interação.

6.2 EXPERIMENTOS E RESULTADOS:

Quanto aos resultados específicos obtidos a partir dos experimentos realizados, pode-se destacar que:

- o primeiro experimento consistiu na geração da superfície topográfica. O programa ***Triangle***, que foi usado para fazer a tringulação de Delaunay, apresentou um desempenho que foi considerado aceitável para um conjunto de 13120 pontos e os recursos computacionais disponíveis. Embora esta quantidade de pontos não possa ser considerada como um grande volume de dados, não foi observada nos manuais deste programa qualquer restrição quanto à quantidade de pontos a serem triangulados.

Além disto, deve-se lembrar que este programa é livre de taxa e tem o código fonte aberto;

- como o programa ***Triangle,*** originalmente, não utiliza interface gráfica para ser executado, foi necessário implementar uma classe chamada "Triangle", que tem a função de passar os dados selecionados para gerar a triangulação e chamar diretamente o programa ***Triangle***, usando para isto uma interface gráfica;
- os resultados obtidos com o desenvolvimento de uma metodologia própria para gerar a triangulação de Delaunay foram importantes para se dominar, conceitualmente, o método de triangulação;
- para visualizar a superfície topográfica foi utilizado o padrão VRML e as classes GeoVRML. Pode-se constatar que com os recursos computacionais disponíveis foi difícil realizar o processo de interação usando-se os controles do navegador VRML e o mouse, para o conjunto de 13120 pontos. Particularmente, isto é crítico em termos de atualização da imagem quando são envolvidas as operações de rotação, translação e redução ou ampliação;
- para desenvolver e implementar os métodos da classe "Retangulo", "Poligono" e "AreaLevantamento" ("disjunto", "toca", "adjacente", "igual", "intercepta", "coincide", "contém" e "está contido") foi necessário selecionar um conjunto menor de dados chamado de região "A". Este conjunto foi totalmente reeditado para eliminar inconsistências no valor de algumas altitudes. Para este conjunto foi idealizado um particionamento em que, simulou-se uma série de relacionamentos espaciais que podem ocorrer à medida que um novo levantamento é obtido;
- a partir da região "A" foi gerado um outro conjunto menor de dados chamado região "B". Com este foi possível simular-se a variação temporal para a superfície topográfica e implementar-se os métodos da classe "Recobrimento" que determinam a variação temporal e geram uma

animação segundo o padrão VRML;

- para visualizar a variação temporal da superfície topográfica usando o conjunto de classes GeoVRML foi necessário implementar um interpolador de coordenadas semelhante ao que existe para interpolar posição. O nome atribuído a esta classe foi "GeoCoordinateInterpolator" e ela tem que ser adicionada ao conjunto de classes GeoVRML para que a animação possa ser visualizada.

Além dos resultados práticos alcançados com o desenvolvimento desta pesquisa, pode-se dizer que a maior contribuição que se está dando é com relação à uma nova estruturação conceitual, que permite representar fenômenos terrestres por meio de um modelo espaço-temporal que era o objetivo principal com este trabalho.

6.3 PERSPECTIVAS PARA FUTUROS TRABALHOS:

A criação de modelos é um assunto que oferece uma gama rica de possibilidades para a pesquisa. Visando dar continuidade aos trabalhos já realizados, sugere-se que:

- realize-se a integração dos modelos de visualização (CAMARGO, 2003) e de persistência (LIMA, 2004) ao modelo de representação, tendo em vista o desenvolvimento de um sistema próprio;
- implementem-se a classe "CorpoHidrico" e algumas de suas especializações, bem como as classes: "Via de Acesso", "Imovel", "Afloramento Rochoso", "Solo", "Deposito Superficial" e "Cobertura Vegetal";
- construa-se uma base de dados com uma maior variedades de fenômenos terrestres;
- estenda-se o conjunto de métodos que foi implementado para lidar com os relacionamentos entre objetos "Poligono" ("disjunto", "toca", "adjacente", "igual", "intercepta", "coincide", "contem" e "este contido") para tratar relações entre outros fenômenos terrestres que

não a superfície topográfica.

- otimizem-se as funções para manipulação e visualização de fenômenos terrestres e se implementem funções voltadas à generalização cartográfica;
- desenvolvam-se novas interfaces e que usem-se outros dispositivos, que não o mouse e teclado, para interação e visualização 3D;
- Estenda-se o modelo proposto para representar elementos temáticos.

7 REFERÊNCIAS

ALHIR, S. S. **UML in a nutshell: a desktop quick reference.** 1. ed. New York: O'Reilly & Associates. 1998.

BARROS, A. J. P. de, LEHFELD, N. A. de S. **Fundamentos de Metodologia: um guia para a iniciação científica**. São Paulo: Mc Graw-Hill, 1986.

BONHAM-CARTER, G. F. Geographic Information Systems for Geoscientists: Modelling with GIS. **Computer Methods in the Geosciences.** v. 13. Ontario: Pergamon, 1994.

BOOCH, G. **Object-Oriented Analysis and Design with Applications.** Redwood City, CA. 1994.

BOOCH, G.; RUMBAUGH, J. ; JACOBSON, I. **The Unified Modeling Language User Guide.** Reading, MA.: Addisson-Wesley. 1999.

BURROUGH, P. A. **Principles of geographical information systems for land resource assessment.** Monographs on soil and resources survey n.12. Oxford: Clarendon press. 1986.

CAMARGO, N. F. **VFT: Uma Estrutura de Aplicação para Visualização de Fenômenos Terrestres em 3D Via Web.** Curitiba, 2003. 182 f. Dissertação (Mestrado em Informática) – Setor de Ciências Exatas, Universidade Federal do Paraná.

CAREY, R. ; BELL, G. **The Annotated VRML97 Reference Manual.** Disponível em: <http://accad.osu.edu/~pgerstma/class/vnv/resources/info/AnnotatedVrmlRef/Book.html> Acesso em: 22 maio 2004.

CHAN, M. C.; GRIFFITH, S. W.; IASI, A. F. **JAVA 1001 Dicas de Programação.** Trad. Miguel Cabrera Fernandes. São Paulo: Makron Books, 1999.

DEITEL, H. M.; DEITEL, P. J. **Java Como Programar.** Trad. Edson Furnankiewicz. 3. ed. Porto Alegre: Bookman, 2001.

DEPARTAMENTO DE GEOMÁTICA. **Mapeamento da Cidade Universitária.** Universidade Federal do Paraná. Curitiba, 2001. 1 CD-ROM.

ERIKSSON, H. E.; PENKER, M. **UML toolkit**. New York: John Wiley & Sons. 1998.

FISHER, P. F. Models of Uncertainty in Spatial Data. In: Longley, P. et al. (eds) **Geographical Information Systems: Principles and Technical Issues.** 2. ed. New York: John Wiley & Sons. v. 1, 1999. p. 191-205.

FLANAGAN, D. **JAVA O Guia Essencial.** Trad. Kátia Roque 3. ed. Rio de Janeiro: Editora Campus, 2000.

GOSLING, J; MCGILTON, H. **The Java Language Environment A White Paper** (may 1996). Disponível em: <http://java.sun.com/docs/white/langenv/index.html > Acesso em 22 maio 2004.

HUTCHINSON, M. F.; GALLANT, J. C. Representation of terrain. In: Longley, P. et al. (eds) **Geographical Information Systems: Principles and Technical Issues.** 2. ed. New York: John Wiley & Sons. v. 1, 1999. p. 105-124.

KEATES, J. S. **Cartographic design and production.** 1. ed. Essex: Longman. 1980.

KJELL, B. **Introduction to Computer Science using Java.** Disponível em: <http://chortle.ccsu.ctstateu.edu/cs151/Notes/chap05/ch05_5.html> Acesso em 22 maio 2004.

KRAAK, M. J.; ORMELING, F. J. **Cartography: Visualization of Spatial Data.** Essex: Addison Wesley Longman. 1996.

LAURINI, R.; THOMPSON, D. **Fundamentals os spatial information systems.** The APIC series n. 37. London: Academic press.1998.

LEMAY, L., COUCH, J.; MURDOK, K. **Lemay's Web Workshop 3D Graphics & VRML 2.0**. Disponível em: <http://docs.rinet.ru/MultiG/> Acesso em: 22 maio 2004.

LIMA, J. D. **Uma Proposta de Interface para a Persistência de Fenômenos Terrestres Representados como Objetos** Curitiba, 2004. 104 f. Dissertação (Mestrado em Informática) – Setor de Ciências Exatas, Universidade Federal do Paraná.

LINDHOLM, T.; YELLIN, F. **The Java Virtual Machine Specification.** Disponível em: <http://java.sun.com/docs/books/vmspec/html/Introduction.doc.html> Acesso em 22 maio 2004.

LONGLEY, P.; GOODCHILD, M.; MAGUIRE, D.; RHIND, D. Introduction. In.: Longley, P. et.al. (eds) **Geographical Information Systems: Principles and Technical Issues.** 2. ed. New York: John Wiley & Sons. v. 1, 1999. p. 1-20.

MAGUIRE, D. J.; DANGERMOND, J. The functionality of GIS In.: **Geographical Information Systems: Principles and Applications.** Maguire, D. J. ; Goodchild, M. F.; Rhind, D. Essex: Longman, London Scientific & Technical. V. 1, 1991. p. 319-35.

MICROSOFT COORPORATION. **Microsoft Developer Studio 97 Visual C++ versão 5.0.** 1997. 1 CDROM.

MOLENAAR, M. **An introduction to the theory of spatial object modelling for GIS.** Taylor & Francis. London. 1998.

NADEAU, D. VRML97: Introduction to VRML 2.0. Disponível em: <http://www.sdsc.edu/~nadeau/Courses/VRML97/> Acesso em: 22 maio 2004.

OKABE, A.; BOOTS, B. ; SUGIHARA, K. **Spatial Tesselations: Concepts and Applications of Voronoi Diagrams.** West Sussex, England: John Wiley & Sons. 1995.

OMG. **Oject Managment Group**. Disponível em: <http://www.omg.org/technology/documents/formal/unified_modeling_language.htm> Acesso em: 1 jun. 2001.

OpenGIS Consortium. **OpenGIS Abstract Specification Topic 5: Abstract Specification Overview** (1999). Disponível em: http://www.opengis.org/docs/99-105r2.pdf Acesso em: 22 maio 2004.

PARALLELGRAPHICS. **Cortona User's Guide - Help online.** 2004.

PEUQUET, D. Time in GIS and geographical dababases. In: Longley, P. at al. (eds) **Geographical Information Systems: Principles and Technical Issues.** 2. ed. Essex: John Wiley & Sons. v. 1, 1999. p. 91-103.

PEUQUET, D. A Conceptual Framework and Comparison of Spatial Data Models. **Cartographica**, v. 21, n. 4, p. 66-113, 1984.

PRICE, R.; SRINIVASAN, B.; RAMAMOHANARAO, K. **Extending the Unified Modeling Language to Support Spatiotemporal Applications.** Asia Techonology of Object Oriented Languages and Systems, p.163-174, 1999.

PRICE, R.; TRYFONA, N.; JENSEN, C. S. Extended Spatiotemporal UML: motivations, requeriments, anda constructs. In.: **Jounal of Database Management**, v. 11, n. 4, p.1-34, 2000.

RHYNE, T. M. **A Commentary on GeoVRML: A Tool for 3D Representation of GeoReferenced Data on the Web** (1999)**.** Disponível em: <http://www.siggraph.org/~rhyne/carto/3D/3D-geovrml.html> Acesso em: 22 maio 2004.

SCHNEIDER, D. K.; MICHIELLOt T, S. M. **VRML Primer and Tutorial**. Disponível em: <http://tecfa.unige.ch/guides/vrml/vrml97/spec/part1/concepts.html#4.3.5> Acesso em: 22 maio 2004.

SHEWCHUK, J. R. **TRIANGLE: A Two-Dimensional Quality Mesh Generator and Delaunay Triangulator.** Disponível em: <http://www-2.cs.cmu.edu/~quake/triangle.html> Acesso em: 22 maio 2004.

SOWIZRAL, H.; NADEAU, D. **SIGGRAPH 99: Introduction to Programming with Java 3D.** Disponível em: <http://www.sdsc.edu/~nadeau/Courses/Siggraph99/>Acesso em: 22 maio 2004.

SUN. **Using NetBeans IDE**. Disponível em:<http://www.netbeans.org/kb/using-netbeans/35/index.html> Acesso em 23 maio 2004.

TOGETHER. **Getting Started Guide - Help online**. 2004.

USGS. United States Geological Survey. American National Standard for Information Systems. **Spatial Data Transfer Standard**. Disponível em: <http://mcmcweb.er.usgs.gov/sdts/SDTS_standard_nov97/part1bo8.html> Acesso em: 22 maio 2004(a).

USGS. United States Geological Survey. **Thematic Mapper (TM).** Disponível em: <http://edc.usgs.gov/products/satellite/tm.html> Acesso em: 22 maio 2004(b).

VIEIRA, A. J. B.; CARVALHO, C. A. P. de. Dados espaciais segundo uma hierarquia de classes e objetos. COLÓQUIO BRASILEIRO DE CIÊNCIAS GEODÉSICAS, 2., 2001, Curitiba. **Anais...** Curitiba: Curso de Pós Graduação em Ciências Geodésicas da Universidade Federal do Paraná. 2001. p. 48-49. Resumos.

VIEIRA, A. J. B.; CARVALHO, C. A. P. de; SLUTER, C. R. Modelagem Espaço-temporal de Fenômenos Topográficos: uma revisão didática. In.: SIMPÓSIO BRASILEIRO DE GEOMÁTICA, 1., 2002, Presidente Prudente. **Anais...** Presidente Prudente: Departamento Cartografia - Faculdade de Ciências e Tecnologia da Universidade Estadual Paulista. 2002. 1 CD-ROM.

VIEIRA, A. J. B. et al. Modelo Espaço-Temporal de Fenômenos Terrestres. In.: COLÓQUIO BRASILEIRO DE CIÊNCIAS GEODÉSICAS, 3., 2003, Curitiba. **Anais...** Curitiba: Curso de Pós Graduação em Ciências Geodésicas e Departamento de Geomática da Universidade Federal do Paraná. 2003. 1 CD-ROM.

Web3D Consortium. Disponível em: <http://www.geovrml.org/> Acesso em: 22 maio 2004.

WORBOYS, M. F. **GIS: a computing perspective.** Taylor & Francis. London. 1995.